Treball Final de Grau

Grau en Enginyeria Informàtica (GEI)

Enginyeria del Software

Disseny i implementació d'una aplicació per l'organització en pisos compartits

Hizzy

"*Living together was never that easy*"

16 de Gener de 2020

Autora: Paula Pallarés Borràs

Directora: María José Casañ Guerrero

FIB UPC UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

Resum

Actualment, i cada cop més, hi ha més gent, especialment a les ciutats, compartint habitatge amb persones alienes al seu cercle i, com és habitual en aquestes situacions, sorgeixen problemes de convivència. Però, què passaria si existís una aplicació capaç d'ajudar a les persones amb aquests problemes? És en aquest àmbit en el qual s'emmarca aquest Treball Final de Grau.

Els problemes de convivència poden venir per diversos motius, ja bé perquè no hi ha una bona relació entre els inquilins per qüestions de caràcter, problemes en la compartició de despeses, o per l'organització interna, és a dir la repartició de tasques relacionades amb la vivenda, la neteja, el compromís i els mals hàbits.

Aquest projecte té l'objectiu de crear una aplicació mòbil que permeti gestionar l'organització interna dins d'un pis compartit, per tal de reduir els problemes el màxim possible i així millorar la convivència entre les persones d'una mateixa vivenda.

Resumen

Actualmente, y cada vez más, hay más gente, especialmente en las ciudades, compartiendo vivienda con personas ajenas a su círculo y, como es habitual en estas situaciones, surgen problemas de convivencia. Pero, ¿qué pasaría si existiera una aplicación capaz de ayudar a las personas con estos problemas? Es en este ámbito en el que se enmarca este Trabajo Final de Grado.

Los problemas de convivencia pueden venir por diversos motivos, ya sea porque no hay una buena relación entre los inquilinos por cuestiones de carácter, problemas en la compartición de gastos, o por la organización interna, es decir el reparto de tareas relacionadas con la vivienda, la limpieza, el compromiso y los malos hábitos.

Este proyecto tiene el objetivo de crear una aplicación móvil que permita gestionar la organización interna dentro de un piso compartido, a fin de reducir los problemas lo máximo posible y así mejorar la convivencia entre las personas de una misma vivienda.

Abstract

Currently, more and more people, especially in cities, are sharing housing with people outside their circle, and, as is the case in these situations, coexistence issues arise. But what if there was an application that could help people with these problems? It is in this area that this Final Degree Project is framed.

Coexistence issues can come for a variety of reasons, either because of a poor tenant relationship due to character issues, cost sharing issues, or internal organisation, such as the sharing of tasks related to housing, cleanliness, commitment and bad habits.

This project aims to create a mobile application that allows managing the internal organisation inside a shared flat, in order to reduce the problems as much as possible and thus improve the coexistence between people in the same home.

Agraïments

A les meves companyes de pis per inspirar-me en la idea i l'ajuda que m'han donat durant el desenvolupament.

A la meva ponent María José Casañ per la seva pasciència i la guia que m'ha proporcionat.

Als meus companys de feina que m'han ensenyat moltes coses i m'han ajudat sempre que ho he necessitat.

I finalment, als meus amics i família per fer-me costat i aguantar-me.

Continguts

1. Introducció

Un dels problemes actuals amb els quals ens trobem els que compartim pis són els conflictes que sorgeixen durant la convivència en un pis compartit. Tota persona que n'hagi compartit o estigui compartint-ne actualment, haurà tingut mínim algun problema amb els seus companys. Jo mateixa estic vivint aquesta situació, per tant ho puc confirmar.

Aquests problemes poden venir per diversos motius, ja bé perquè no hi ha una bona relació entre els inquilins per qüestions de caràcter, per compartiment de despeses, o per l'organització interna, és a dir la repartició de tasques relacionades amb la vivenda, la neteja, el compromís i els mals hàbits. Aquests últims són molt importants, ja que acaben repercutint en la qualitat de les relacions entre persones que conviuen en un pis compartit.

A més, cada cop és més comú el fet de compartir habitatge, ja que al final és una manera més barata de poder viure, especialment a les ciutats on el cost del lloguer és molt alt. Els preus dels pisos de lloguer cada cop són més elevats a causa de la gran demanda que hi ha i oferta limitada. Això provoca que moltes persones no es puguin permetre pagar un lloguer complet per si soles com bé es mostra a la Figura 1, on la principal raó per compartir vivenda és no poder pagar un lloguer sencer.

Figura 1: Percentatges de les raons per compartir habitatge [1]

Els articles de El Economista [1] i El Diario [2], basats amb estudis recents, expliquen que en tan sols un any la mitjana d'edat dels espanyols que comparteixen pis ha passat dels 29 als 34 anys. Avui dia només el 10% del mercat de lloguer són estudiants.

Aquest mercat està en creixement i cada cop hi ha més gent que comparteix vivenda. Si tenim en compte que sovint acaben havent-hi problemes de convivència, què podem oferir per solucionar aquesta problemàtica i reduir-los? S'ha fet una cerca de les solucions TIC existents (vegeu apartat 4) i s'ha plantejat a partir d'aquí una proposta que ajudi a millorar la convivència dins de pisos compartits. Aquesta solució es presenta com una App gamificada destinada a millorar la convivència dins d'un pis compartit. L'App vol ajudar a millorar l'organització interna de les persones que comparteixen pis.

Aquesta nova aplicació rep el nom de Hizzy. Hizzy és una forma informal de dir casa amb anglès i com que el lema de l'aplicació és "*living together was never that easy*" em va semblar un nom adequat.

Segons la normativa del TFG en el Grau d'Enginyeria Informàtica [3] aquest projecte correspon a la modalitat A projectes realitzats a la UPC i pertany a la titulació Grau d'Enginyeria Informàtica amb especialitat d'Enginyeria del Software.

2. Actors

Desenvolupadora

La desenvolupadora, Paula Pallarés Borràs, és qui s'encarrega de dur a terme totes les tasques associades a aquest projecte, el desenvolupament del codi, els testos i la documentació requerida.

Directora del projecte

La directora del projecte, María José Casañ Guerrero, s'encarregarà de guiar i ajudar a la desenvolupadora en els possibles obstacles que es pugui trobar, i a més, supervisarà el projecte.

Beneficiaris

El producte anirà dirigit a qualsevol persona que convisqui amb altres persones dins d'un mateix habitatge. Per tant, els beneficiaris seran els mateixos usuaris de l'aplicació. El producte està pensat perquè l'utilitzin d'igual manera i en el mateix període de temps les persones que conviuen en un mateix pis. I a més, la desenvolupadora pot ser també una possible usuària de l'aplicació.

Companyes de pis

Les companyes de pis de la desenvolupadora, donaran *feedback* sobre el projecte per tal de millorar-lo.

3. Conceptes inicials

En aquesta secció es descriuran els conceptes que són necessaris entendre per poder explicar correctament el projecte que es desenvoluparà en aquest Treball Final de Grau.

3.1. Organització interna

Com s'ha explicat a la introducció d'aquest projecte, l'àmbit en el qual se situa aquesta aplicació és l'organització interna d'un pis compartit. Així doncs, s'ha d'explicar que entenem com a organització interna i en quins casos podria provocar una problemàtica.

Entenem organització interna com l'organització de tasques dins un habitatge entre els seus inquilins. Aquestes tasques poden ser tasques de neteja com per exemple netejar la cuina o el bany. O també poden ser d'altres tipus com per exemple anar a comprar o anar al banc a pagar el lloguer.

Segons com s'organitzin aquestes tasques es pot produir una problemàtica, ja sigui perquè hi ha persones que no realitzen el que se'ls ha assignat o si ho fan, és dies més tard. O bé perquè el que s'ha fet, no ha sigut de la forma que s'esperava i s'ha realitzat malament, com per exemple netejar molt ràpidament i malament el bany deixant-lo brut igualment.

Altres aspectes que entren dins de l'organització interna són la distribució de tasques, de la qual hi ha diferents variants que he extret a través de parlar amb altra gent que es troba en aquesta situació. Les dues formes més comunes són: distribuir les tasques d'una forma fixa és a dir, la mateixa persona s'encarrega sempre del mateix i l'altra forma és que aquesta distribució vagi rotant, de manera que cada persona acaba realitzant totes les tasques en un cert període de temps. Això significa que cada setmana van canviant les assignacions, si per exemple aquesta setmana has de netejar la cuina, la setmana que ve et tocarà netejar el lavabo, i l'altra el menjador i després la cuina de nou, i així consecutivament.

3.2. Gamificació

Com s'ha descrit a la introducció d'aquest document, el projecte consisteix a desenvolupar una aplicació gamificada. Per tant, cal explicar què és gamificació.

Aquest concepte prové de la paraula anglesa *Game* que significa joc. Gamificació [17] és l'ús de mecàniques de joc en entorns i aplicacions no lúdiques. Aquestes tècniques s'utilitzen amb la finalitat de potenciar la motivació, la concentració, l'esforç, la fidelització i altres valors comuns entre els jocs. Generalment s'utilitza per influir i motivar grups de persones.

En aquest projecte s'utilitzarà de dues formes. La primera consisteix en un sistema de puntuació sobre les tasques de la casa. Les tasques estaran distribuïdes setmanalment i repartides entre els integrants d'una casa equitativament. Cada tasca dóna un màxim de 10 punts a l'usuari que l'hagi completat, 5 punts un cop s'hagi completat i 5 punts d'una valoració mitjana entre la resta de companys. És a dir, si un company completa una tasca, la resta hauran de valorar la feina feta en una puntuació entre 0 i 5. Aquest sistema de puntuacions generarà un rànquing setmanal entre els integrants, fent que qui tingui més punts estigui en primera posició. També hi haurà tasques extres creades per l'usuari que atribuiran 2,5 punts a qui les hagi completat. Si la data de finalització d'una tasca és anterior al dia que s'ha completat s'aniran descomptant 0,5 punts per dia de retràs dels 5 punts màxims que es poden obtenir per completar una tasca.

La segona consisteix en una tècnica utilitzant premis i càstigs. De forma que si hi ha una persona que durant tres setmanes seguides ha estat l'últim del rànquing, se li atribuirà un càstig escollit entre els companys. Aquest càstig haurà de ser acceptat per la persona castigada, si no és així, tindrà un càstig pendent. El mateix passa amb el premi, si hi ha un usuari que durant cinc setmanes seguides ha estat en primera posició al rànquing setmanal, tindrà un premi pendent que també s'escollirà entre els companys. Aquest premi també haurà de ser acceptat per l'usuari premiat.

Aquestes tècniques pretenen motivar a les persones que comparteixen pis a través de la competitivitat i de les bonificacions. Intentant així, aconseguir el compromís necessari per tenir una bona harmonia, en el sentit que cadascú sigui responsable i realitzi les tasques de forma correcta en les dates acordades.

4. Estat de l'art

Per justificar si és convenient aprofitar i adaptar una solució existent o si s'ha de dissenyar-ne una de nova, he decidit fer un estudi de mercat. Per fer-lo, he analitzat les diferents aplicacions que hi ha relacionades sobre el tema de compartir pis [4] [5]. He trobat tres tipus d'aplicacions.

4.1. Cercadors

Aquest tipus d'aplicacions estan enfocades en el tema de trobar pis o habitació disponible segons les necessitats de l'usuari. El mercat en aquest tema sembla que està bastant cobert, per tant hi ha moltes aplicacions que ajuden als seus usuaris a trobar un pis segons la localització i el preu. Com per exemple Habitaclia, Fotocasa, Idealista, etc. Però jo m'he centrat en aquelles que tenen en compte el real problema que he plantejat abans, la convivència.

Badi

Badi [6] és una aplicació que opera internacionalment on el factor principal per buscar pis són els futurs companys. Hi ha dos tipus d'usuaris, els que busquen i els que ofereixen. Els usuaris que busquen pis s'han de crear un perfil amb les seves dades personals i a més, amb dades referents als interessos i maneres de ser. Mentre que per altra banda, els usuaris que ofereixen lloguer d'habitacions, a més de la informació personal i interessos, han d'afegir també quin és el perfil de persona que busquen. Així doncs, Badi connecta als seus usuaris entre ells de forma que siguin compatibles.

També té un sistema de geolocalització per trobar habitacions tenint en compte la proximitat, és a dir, el segon factor és la proximitat, i inclou un xat intern perquè els seus usuaris es puguin comunicar un cop haver connectat.

També l'anomenen el "Tinder dels companys de pis" perquè tracta de fer aquesta connexió entre persones.

Habitoom

Habitoom [7] és una *WebApp* que opera internacionalment on també hi ha dos tipus d'usuaris que s'han de crear un perfil detallat per tal de connectar-los entre ells com fa Badi. Però, a més de connectar persones que tenen els mateixos interessos i gustos, també té en compte la zona de la ciutat on és millor que visquin

els seus usuaris amb acord a les seves necessitats. Per exemple, si t'agrada molt la natura, et buscarà també habitatge a una zona amb molta natura.

A més, en la descripció dels pisos també explica com és la sensació de viure allà pels que ja estan vivint o han viscut. Per tant, els usuaris que busquen pis es poden fer una idea de com seria viure-hi.

4.2. Aplicacions de despeses compartides

En aquests tipus d'aplicacions el tema principal són els diners i la divisió de despeses en comptes comuns. En aquest cas el mercat també és molt ampli, ja que és un dels problemes que més afecta en el fet de compartir pis. També he decidit agafar les més populars.

Splitwise

Splitwise [8] és una *WebApp* que opera internacionalment que s'encarrega de mantenir un registre de les despeses compartides que tenen els seus usuaris amb els seus companys de pis, companys de viatge, parella, amics, família o altres grups. El que permet aquest registre és tenir diferents grups on dins de cadascun hi ha una llista de qui paga que i quant ha costat. D'aquesta forma es calcula quant deuen els seus usuaris i a qui, i quant els hi deuen i qui dins d'aquell grup. A més, ofereix una vista general on els usuaris poden veure quant deuen en total sumant tots els grups i quant els deuen, per tenir un balanç real dels diners.

A més d'afegir el que s'han gastat i com, també es pot escollir la manera de dividir aquells diners: de forma igual entre tots els integrants del grup, només alguns dels integrants, etc. També permet que la persona a la qual li deuen diners pugui reclamar-los enviant notificacions als que li deuen.

Un cop els usuaris han pagat a qui li devien diners, afegeixen un nou registre indicant que ja s'ha liquidat aquell deute si el pagament ha estat extern a l'aplicació, com per exemple en efectiu, o el registre es fa automàtic si utilitzes un mètode de pagament online que ofereix l'aplicació.

També permet als seus usuaris no tenir la necessitat de crear un grup si per exemple han anat a sopar d'una manera eventual amb un grup de persones amb les quals no solen freqüentar. És a dir, amb aquelles persones amb les quals no tindran altre tipus de despeses comunes.

Finalment, també té opcions de diferents monedes. Si per exemple els usuaris han anat de viatge a Londres i han pagat algunes coses amb lliures i altres amb euros també els permet fer aquesta barreja de divises.

Tricount

Tricount [9] és una *WebApp* molt similar a Splitwise que també es dedica a mantenir un registre de les despeses compartides en grups. El que les diferencia més és que en Tricount no es necessita un perfil per formar part d'un grup. A més, no permet fer barreja de divises i sempre ha d'existir un grup per tenir en compte les despeses comunes. No permet esborrar un integrant del grup si aquest encara té deutes pendents.

Com que no es necessita un perfil es poden invitar als altres integrants del grup per qualsevol xarxa social. Les despeses es poden ordenar per títol, quantitat, data o persona que ha pagat.

Finalment, es poden afegir imatges a les despeses, cosa que és molt útil per adjuntar el tiquet i evitar confusions.

4.3. Aplicacions d'organització interna

Pel que fa a aquest tipus d'aplicacions hi ha moltes però estan enfocades en l'organització de tasques per tota classe de situacions com per exemple un projecte, llistes de tasques personals, etc. Aplicacions com Trello o Wunderlist es dediquen a això, que també es podria utilitzar en un pis compartit però no acaben tenint l'enfocament que jo busco. Jo busco aplicacions dedicades exclusivament per millorar la convivència en una mateixa casa. Vaig veure aplicacions com Homeslice i Chorma que es dedicaven a aquest tema però per algun motiu ja no estan disponibles. Així i tot, hi ha altres aplicacions que tenen un enfocament semblant.

Picniic

Picniic [10] és una *WebApp* enfocada per famílies que ofereix organització en totes les activitats, tasques i informació de la família. També conté un localitzador a temps real de la localització de cada membre de la família que ajuda a saber en tot moment on se situen els integrants. A més, té un calendari comú per tots on es poden posar tots els plans futurs o activitats importants.

Es poden crear tasques i assignar-les entre els membres de la família i ofereix també un control parental sobre l'accés a internet. A més, es poden adjuntar fitxers com per exemple arxius mèdics o informació important de la casa. També es poden crear petits subgrups per tenir altres integrants com pot ser una cangur i només donar-li accés a allò que li pertoca.

Es poden afegir receptes de cuina i qualsevol altre tipus de tradició familiar, i a més també es poden afegir fotos i compartir-les amb els membres de la família.

Conté un xat intern que permet està connectat i al dia de tots els missatges que s'envien.

Finalment, es poden crear llistes de la compra.

Tody

Tody [11] és una *App* enfocada en la neteja que utilitza la motivació personal per ajudar als seus usuaris en la tasca de netejar la casa. Per motivar als seus usuaris té un calendari on poden escollir un *planning* estimat de quins dies de la setmana haurien de netejar. Tody manté un registre de quan l'usuari acaba netejant realment i segons el temps que ha trigat té un indicador de com brut és l'usuari. Com més triga a netejar i més se'n va del *planning* estimat, més brut és. Això ho té amb una barra de colors que com més net més verd i com més brut més vermell de tal forma que provoca que l'usuari es motivi a arribar sempre al color verd.

Tot això ho permet compartir també amb un grup i assignar les tasques de neteja a persones diferents. Això significa que els companys també podran veure la netedat de l'usuari.

És una aplicació que s'obté pagant.

4.4. Conclusió

Un cop analitzat l'estat de l'art pel que fa a les aplicacions destinades a millorar la convivència en un pis compartit, en podem extreure diverses conclusions:

Per una banda, el tema que queda més vacant és el de l'organització interna, ja que no existeix una aplicació enfocada només a això, en la que puguis controlar o motivar als usuaris a portar una bona organització dins d'un habitatge.

Per altra banda, tampoc existeix una aplicació que utilitzi la gamificació per assegurar el compromís dels integrants de la casa.

A més, el tema dels diners sembla molt cobert i complet amb les aplicacions existents d'igual manera com el dels cercadors. Per tant, no els inclouré en aquest projecte, ja que m'agradaria que la meva aplicació aportés un valor afegit innovador.

Tenint en compte aquests fets, s'ha decidit crear una aplicació gamificada que es dediqui exclusivament a millorar la convivència en un pis compartit enfocada amb l'organització. De les eines estudiades, no hi ha cap que incorpori funcionalitats per millorar l'organització i per motivar als usuaris de les vivendes compartides a fer les tasques pertinents.

5. Abast

5.1. Objectius principals

Com s'ha explicat anteriorment, l'objectiu principal d'aquest projecte és dissenyar i desenvolupar una aplicació destinada a millorar la convivència dins d'un pis compartit oferint una eina per millorar l'organització interna, les tasques i la planificació, utilitzant la gamificació com a mètode de motivació i compromís. Per tal complir aquest objectiu s'hauran de dur a terme una sèrie d'objectius específics:

- Que l'aplicació permeti tenir grups i poder afegir als companys de pis a un grup concret.
- Que l'aplicació permeti crear tasques i llistes per poder portar un control de les tasques que s'han de fer al pis.
- Que l'aplicació permeti assignar tasques per tal que els integrants del grup les facin i els companys les puguin valorar.
- Que l'aplicació ofereixi dos tipus de mètodes base d'organització: el fix i el rotatori. En el fix durant un període de temps els usuaris sempre es dediquen a fer les mateixes tasques. En el rotatori a partir d'un temps les tasques van alternant d'usuaris, per exemple cada setmana realitza la tasca algú diferent.

  Els mètodes són personalitzables. Això implica que un mètode té usuaris, que no han de ser els mateixos que els del grup, ja que si hi ha una temporada en la qual un integrant no viu al pis es puguin modificar els usuaris.

  D'igual forma passa amb les tasques que pertanyen a un mètode, imagina que durant una temporada no s'ha de realitzar una tasca o una tasca canvia segons l'època de l'any en la qual et situïs. S'han de poder modificar aquestes tasques.
- Que l'aplicació permeti completar les tasques per indicar que ja estan fetes.
- Que l'aplicació ofereixi premis i penalitzacions per aquelles situacions en les quals els companys ho considerin, com per exemple el fet de no realitzar una tasca quan toca.
- Que hi hagi un sistema de puntuacions, explicat anteriorment, que permeti tenir un rànquing per utilitzar la competitivitat com a motivació personal.

5.2. Objectius secundaris

Un cop definits aquests objectius, es poden extreure un conjunt de subobjectius que caldrà completar per aconseguir els objectius principals:

- Que els usuaris puguin valorar les tasques realitzades pels seus companys per si una tasca es fa, però malament. Això permetria que fos més fàcil d'aplicar el mètode dels càstigs.
- Que els usuaris puguin escollir els premis o els càstigs quan tinguin en tinguin un pendent.
- Poder tenir incorporat un calendari per fixar els terminis de realització de tasques i un sistema per a notificar als usuaris.

- Oferir tasques extres per obtenir punts extres.
- Oferir usuaris administradors de grups perquè realitzin aquelles funcionalitats que requereixin líders.

5.3. Altres requisits

Es vol dissenyar una aplicació atractiva per als usuaris, usable i multiidioma, ja que per exemple hi ha molts estudiants d'*erasmus* que també comparteixen pis.

5.4. Riscos

Planificació

Un dels possibles riscos és no fer una bona planificació temporal, ja que en ser un Treball de Final de Grau hi ha una data límit i potser no em dóna temps de fer totes les funcionalitats. Per evitar la mala planificació és molt important l'ajuda de la directora del projecte.

Disseny

Per no tenir problemes durant la implementació de l'aplicació, cal fer un bon disseny pensat per optimitzar al màxim les funcionalitats definides als requisits. Per evitar-ho també serà necessària l'ajuda de la directora.

Mal ús de l'aplicació

És possible que els usuaris utilitzin l'aplicació d'una manera no adequada com per exemple escollir càstigs molt dolents o fer la punyeta entre els integrants. Per tal d'evitar-ho hi haurà una selecció restringida i l'usuari castigat haurà d'acceptar el càstig.

Diferent compromís entre els integrants

Un altre problema podria ser que entre els diferents companys de pis hi hagi un nivell diferent de compromís, és a dir, que hi hagi persones que l'utilitzin i li donin importància i d'altres que no li'n donin. Per evitar-ho caldrà incrementar la motivació personal a través de la competitivitat i la gamificació.

6. Metodologia

6.1. Metodologia de treball

Per al desenvolupament de l'aplicació s'utilitzarà una metodologia *Agile*, vista durant els cursos del grau, que és una metodologia iterativa i incremental.

L'objectiu d'aquesta metodologia és desenvolupar un sistema a través de cicles repetits (iterativa) i en porcions més petites alhora (incremental). Quan ens referim a iterativa ens referim a les divisions en iteracions en què es divideix el projecte i incremental significa que cada cop es van afegint més funcionalitats a les iteracions fins que el producte està acabat.

Els principals avantatges són la flexibilitat, que permet dimensionar els projectes minimitzant els riscos i facilita la priorització i la presa de decisions.

Es tria aquest tipus de metodologia per la flexibilitat que ofereix en projectes petits on els requisits no estan completament definits des de l'inici o perquè van canviant.

Per tant, dividiré el projecte en *sprints*/iteracions de 10 dies que m'ajudaran a rebre feedback un cop s'acabin, ja que donaré a provar l'aplicació a diferents usuaris per tal d'entendre millor les seves necessitats.

Cada iteració estarà dividida en 5 fases: la planificació, el disseny, el desenvolupament, el *testing* i la revisió de l'*sprint*.

6.2. Eines de seguiment

Per tal de poder fer un bon seguiment del projecte i portar un bon control del desenvolupament, s'utilitzaran les següents eines:

GitHub. Serà l'eina que em permetrà tenir el projecte a un repositori privat i així poder utilitzar el control de versions de Git, tenint un registre de l'evolució del projecte amb les diferents versions.

Trello. Serà l'eina que em permetrà organitzar-me les tasques durant els *sprints*. Tenint en compte els requisits, m'ajudarà a crear un *product backlog* i anar planificant les iteracions a partir d'aquest.

7. Planificació Temporal

El temps estimat de duració d'aquest projecte és al voltant de 4 mesos. Es va començar dilluns 16 de setembre del 2019 i té com a data de finalització 16 de gener de 2020, ja que la defensa del projecte és del 23 de gener. Com que el Treball Final de Grau són 18 crèdits i per cada crèdit s'esperen unes 30 hores, això donaria al projecte una càrrega de treball d'unes 540 hores.

7.1. Descripció de tasques

Tasques Inicials (GEP)

En aquesta fase es farà una visió general del projecte per definir, planificar, fer anàlisi de requisits i avaluar la viabilitat del projecte. Es definirà l'abast i es farà la contextualització, també la planificació temporal i finalment la gestió econòmica i sostenibilitat.

Anàlisi i disseny

En aquesta fase és quan es generarà tota la documentació relacionada amb l'anàlisi de funcionalitats i requisits de l'aplicació. Es crearà el *product backlog* que contindrà totes les històries d'usuari i s'atribuirà un ordre d'importància en les tasques. A més, també es farà el disseny de tota l'aplicació, de l'arquitectura general, disseny del domini, model de base de dades en UML i el disseny de la interfície d'usuari creat amb *mockups*.

Implementació

La tercera fase és la implementació de l'aplicació. Es començarà amb una preparació de l'entorn i les eines per poder realitzar el desenvolupament del projecte.

Aquí es començarà a aplicar la metodologia *Agile*, dividint la fase en sprints de dues setmanes on s'agafaran les històries d'usuari del *product backlog* (segons la importància de les tasques) que es volen desenvolupar i es treballaran.

A l'inici de cada iteració, s'implementarà en primer lloc la part de *backend* i de la lògica del domini. A continuació la de *frontend*, és a dir la interfície gràfica necessària perquè els usuaris puguin dur a terme les funcionalitats.

És possible que quan el desenvolupament del projecte estigui en procés apareguin nous possibles requisits, en cas que això sigui així s'estudiarà la rellevància de cadascun d'ells i en cas que s'arribi a la conclusió que són necessaris, s'adaptaran els *sprints* següents, per tal d'afegir noves funcionalitats sense necessitar més temps de l'establert.

Al final de cada iteració es farà una part de *testing* així com un *review* de com ha anat l'*sprint* per poder prendre decisions sobre els següents.

Documentació

Durant l'última fase es recollirà tota la documentació generada durant el desenvolupament del projecte i es complementarà amb la informació que sigui necessària. També es farà un manual d'usuari per l'aplicació. A més, al final de la fase es prepararà la defensa del projecte.

7.2. Recursos

Per tal de dur a terme el desenvolupament del projecte, s'utilitzaran els següents recursos:

Recursos personals

Una persona, que dedica unes 4,5 hores al dia al llarg de tot el projecte, que farà la planificació, anàlisi, disseny, desenvolupament i test de l'aplicació. Aquesta persona actuarà com diversos rols diferents: Cap de projecte, analista, dissenyadora, programadora i tester.

Recursos hardware

- **Portàtil Macbook Pro 2014 15":** ordinador des del qual es desenvoluparà el projecte.
- **Iphone 7 (Apple):** mòbil amb sistema operatiu iOS per poder testejar l'aplicació.
- **Xiaomi Redmi 8A (Xiaomi):** mòbil amb sistema operatiu Android per poder testejar l'aplicació.

Recursos software

- **Visual Studio Code:** Editor de text que s'utilitzarà per desenvolupar tota la part de *frontend* del projecte.
- **Pycharm:** Editor de text que s'utilitzarà per desenvolupar tota la part de *backend* del projecte.
- **React native:** *Framework* que s'utilitzarà per realitzar el *frontend*.
- **Heroku Cloud Platform:** Plataforma per tenir la *url* de la API i la base de dades.
- **Github:** Eina que contindrà el repositori del projecte.
- **Gitkraken:** Controlador de versions per pujar codi a un repositori de Github.
- **Lucidchart:** Eina que s'utilitzarà per crear l'UML (Unified Modeling Language).
- **Figma:** Eina que s'utilitzarà per crear els *mockups* de l'aplicació.
- **Swagger:** Editor de text que s'utilitzarà per documentar els *endpoints* a desenvolupar i testejar la API.

7.3. Estimació d'hores

Tasca	Hores	Dependències
1. Tasques inicials (GEP)	**61h**	**-**
1.1. Definició de l'abast i contextualització	25h	-
1.2. Planificació temporal	8h	1.1
1.3. Estimació de costos	10h	1.2
1.4. Entrega final GEP	18h	1.3
2. Anàlisi i disseny	**85h**	**-**
2.1. Anàlisi requisits i funcionalitats	35h	1.1
2.2. Disseny arquitectura	15h	2.1
2.3. Disseny UML	10h	2.2
2.4. Disseny interfície	25h	2.3
3. Implementació	**260h**	**2**
3.1. Preparació entorn	10h	2.4
3.2. Implementació *backend*	70h	3.1
3.3. Implementació *frontend*	130h	3.1
3.4 *Testing* i *review*	50h	3.2 i 3.3
4. Documentació	**130h**	**-**
4.1. Manual d'usuari	20h	3.4
4.2. Documentació final	90h	-
4.3. Preparar defensa	20h	4.1 i 4.2
Total	**536h**	**-**

Taula 1: Estimació de temps

5.4. Diagrama de Gantt

Aquest diagrama de Gantt està creat tenint en compte les hores dedicades a cada tasca indicades a l'apartat anterior, també que el projecte està desenvolupat per una sola persona que dedica unes 4,5 hores al dia.

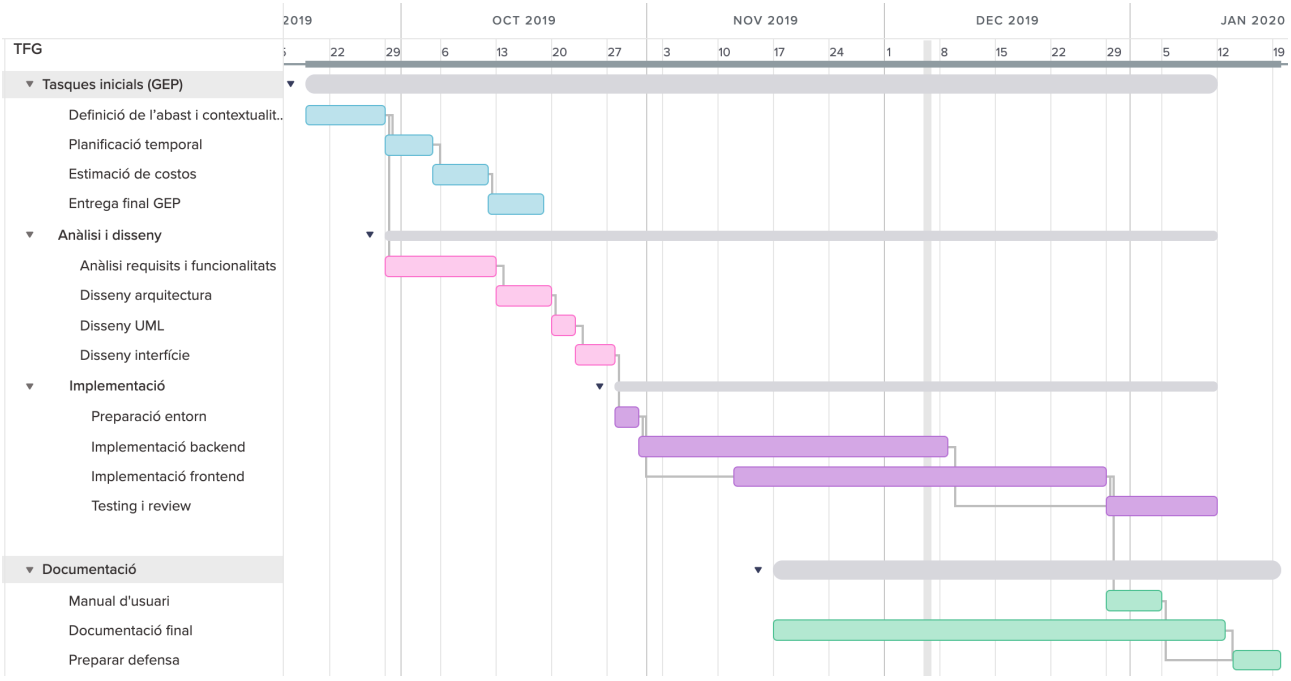


Figura 2: Diagrama de Gantt realitzat amb TeamGantt [13]

Com es pot veure les tasques planificació temporal, estimació de costos i entrega final de GEP es donen a terme paral·lelament amb altres tasques. Per entendre la durada en dies de cadascuna de les tasques és necessari saber que les tasques inicials ocupen més o menys 3 hores diàries, això vol dir que a la segona tasca que es realitza juntament amb GEP se li dediquen unes 1,5 hores diàries.

Per altra banda, també es pot veure que la implementació del *backend* i el *frontend* es fan a la vegada durant uns dies, durant aquest període de temps es dediquen unes 2 hores al *backend* i unes 2,5 hores al *frontend*. I en el cas de frontend, backend i documentació a la vegada seria 1,5 hores de backend 2,5 de frontend i 0,5 de documentació.

8. Gestió de riscos

Poden succeir desviacions de la planificació estimada per diferents motius:

- El temps dedicat a un cas d'ús és més elevat del que s'havia estimat. En aquest cas es reorganitzarà el repartiment de casos d'ús entre les iteracions. Primer s'intentarà reequilibrar les tasques repartides sense eliminar cap funcionalitat. Si això no és possible, es mirarà de simplificar els requisits secundaris, i en cas que això tampoc es pugui dur a terme, s'eliminarà algun requisit secundari. De tota manera, abans de prendre aquesta mesura, cal tenir en compte que s'ha deixat un marge d'hores extres que la desenvolupadora podria dedicar al projecte en cas que fos necessari si hi hagués moltes desviacions, tot i que no s'espera que sigui així.

- El temps dedicat a algun cas d'ús és menor del que s'havia estimat. En aquest cas es continuaria fent les següents històries d'usuari per ordre de rellevància. En cas que estiguin totes les funcionalitats desenvolupades i que s'hagi invertit menys temps del que s'esperava, s'aprofitaria per afegir algunes funcionalitats extra. En cas que s'afegissin, s'hauria de calcular que el projecte pogués estar acabat el dia previst.

- Es podria donar el cas de fer algun canvi amb les eines que s'utilitzaran per a alguna de nova. Això provocaria un temps d'aprenentatge i d'adaptació que s'hauria d'afegir a la planificació. En aquest cas s'hauria de comprovar que el projecte es pogués realitzar en el temps previst. Si no és possible, s'hauria d'intentar evitar introduir aquestes noves eines i continuar amb les planificades.

- També podria passar que el temps dedicat al *testing* d'alguna tasca sigui excessiu. S'hauria de mirar quines tasques són més prioritàries, i a partir d'aquí dedicar menys temps de *testing* a aquelles més simples i més secundàries.

En qualsevol dels casos s'hauria de consultar amb la directora del projecte.

9. Gestió econòmica

9.1. Identificació i estimació dels costos

Per obtenir una estimació econòmica d'aquest projecte cal tenir en compte diversos tipus de costos.

Recursos humans

En primer lloc els recursos humans, que en aquest cas serà una única persona a la qual li recauran tots els rols. Aquests rols es corresponen a Cap de projecte, Analista, Dissenyadora, Programadora i Tester. Per cadascun d'aquests es té en compte el salari definit segons les dades trobades a l'estudi del mercat laboral a Espanya elaborat per Hays [14].

Els sous que es descriuen al document són els següents:

Cap de projecte	Analista	Dissenyadora	Programadora	Tester
27€/h	23€/h	16€/h	15€/h	14€/h

Taula 2: Euros per hora que cobra cada un dels rols

Les hores que dedica cada rol estan basades en la divisió de tasques i el diagrama de Gantt de la planificació temporal del projecte.

Tenint en compte això, la despesa que suposen els recursos humans es desglossa de la manera següent:

Tasca	Cap de projecte	Analista	Dissenyadora	Programadora	Tester	Cost per tasca i fase
Tasques inicials (GEP)	**61h**	**-**	**-**	**-**	**-**	**1647€**
Definició de l'abast	25h	-	-	-	-	675€
Planificació temporal	8h	-	-	-	-	216€
Estimació costos	10h	-	-	-	-	270€
Entrega final GEP	18h	-	-	-	-	486€
Anàlisi i disseny	**10h**	**25h**	**50h**	**-**	**-**	**1645€**
Anàlisi requisits	10h	25h	-	-	-	845€
Disseny arquitectura	-	-	15h	-	-	240€
Disseny UML	-	-	10h	-	-	160€
Disseny interfície	-	-	25h	-	-	400€

Tasca	Cap de projecte	Analista	Dissenyadora	Programadora	Tester	Cost per tasca i fase
Implementació	**5h**	**-**	**-**	**175h**	**80h**	**3880€**
Preparació entorn	-	-	-	10h	-	150€
Implementació *backend*	-	-	-	55h	15h	1035€
Implementació *frontend*	-	-	-	110h	20h	1930€
Testing i *review*	5h	-	-	-	45h	765€
Documentació	**110h**	**-**	**-**	**20h**	**-**	**3270€**
Manual d'usuari	-	-	-	20h	-	300€
Documentació final	90h	-	-	-	-	2430€
Preparar defensa	20h	-	-	-	-	540€
Hores per rol	**186h**	**25h**	**50h**	**195h**	**80h**	**-**
Cost per rol (sense seguretat social)	**5022€**	**575€**	**800€**	**2925€**	**1120€**	**10442€**
Cost per rol*1,35% seguretat social	**6779,7€**	**776,25€**	**1080€**	**3948,75€**	**1512€**	**14096,7€**

Taula 3: Estimació de costos per tasques, rols i hores dedicades

Recursos Hardware

En segon lloc el *hardware* utilitzat, que serà un ordinador portàtil i dos telèfons mòbils amb diferent sistema operatiu (un iOS i l'altre Android), i per tant cal calcular el cost de la part proporcional de l'amortització d'aquests dispositius.

El preu de compra del **Macbook Pro 2015** és de **2185,25€** [15]. Tenint en compte que el temps d'amortització és d'aproximadament 5 anys, cal calcular també que l'any té aproximadament 250 dies laborables i que cadascun dels dies es treballen 8 hores, trobem que el cost d'amortització per hora de treball d'aquest portàtil és d'aproximadament 0,22€. Per tant, si ho multipliquem per les hores totals del projecte (536h) el cost de l'amortització total del portàtil és **117,92€**.

El preu de compra de l'iPhone 7 és de **769€** [15]. Tenint en compte els mateixos paràmetres anteriors excepte que fa 3 anys en comptes de 5, trobem que el cost d'amortització per hora de treball és aproximadament 0,13€. Per tant, si ho multipliquem per les hores dedicades a la programació de frontend del projecte (130h), el preu final és **16,9€**.

El preu de compra del Xiaomi Redmi 6A és de **86,49€** [16]. Igual que anteriorment amb els mateixos paràmetres excepte 1 any de temps d'amortització, trobem que el cost d'amortització per

hora de treball és aproximadament 0,04€. Per tant, si també ho multipliquem per les hores de frontend, el preu final és **5,2€**.

Per tant, el preu total dels recursos hardware és **140,02€**.

Recursos Software

Pel que fa al *software*, tots els programes que s'utilitzaran seran gratuïts per tant no hi haurà costos generats.

Despeses Generals

Respecte a les despeses generals o despeses indirectes, tenim aquelles despeses que no s'han contractat directament per poder desenvolupar el projecte, però que sense elles no es podria realitzar. En aquesta categoria tindríem l'electricitat i l'internet. L'electricitat em costa 0,14€/kW, si ho multipliquem per 536 hores, seria un total de **75,04€**. Pel que fa a l'internet són 30€ al mes per 4 mesos, és a dir **120€**.

Per tant, el preu total de les despeses generals és de **195,04€**.

Contingència

Finalment, es tindrà en compte un 15% de contingència en cas que hi hagués algun problema durant el desenvolupament que pogués provocar un augment del cost. Aquesta contingència només s'aplicaria pels recursos humans en cas que s'haguessin de dedicar més hores al projecte. Per tant, sumaria un total de **2114,51€**.

Total

Tenint en compte el cost de cadascun dels apartats anteriors, el cost final del projecte serà de **12343,36€**.

Tipus de cost	**Preu**
Recursos humans	14096,7€
Recursos *hardware*	140,02€
Recursos *software*	0€
Despeses generals	195,04€
Contingència	2114,51€
Total	**16436,27€**

Taula 4: Taula resum dels costos totals del projecte

9.2. Control de gestió

Per tal de poder obtenir un control, en finalitzar cadascuna de les tasques s'escriurà el temps en hores real que se li ha dedicat a cada tasca. A partir d'aquí, s'agafarà la diferència amb el temps estimat i, en cas que aquesta diferència sigui molt àmplia, s'estudiarà l'origen d'aquest canvi i es reajustaran les fases següents per tal de no cometre el mateix error.

El control serà més exhaustiu en les fases d'anàlisi i disseny i sobretot en la implementació, ja que és on hi ha més risc que es produeixin errors que modifiquin la planificació, i en conseqüència el pressupost. Com ja s'ha comentat a l'apartat de planificació, és possible que sigui necessari fer més hores, però el cost d'aquestes hores ja s'ha afegit a la part de contingència.

Poden passar contingències com per exemple la necessitat de nou material, és a dir recursos hardware. També podria passar que es necessiti algun recurs software de pagament durant el llarg del projecte o en algun punt en concret.

En cas que hi hagi un desviament del cost del projecte, es faran servir els següents indicadors de desviació:

- Desviament recursos humans en sou = (cost estàndard – cost real) * hores reals
- Desviament recursos humans en eficiència = (hores estàndard – hores reals) * sou estàndard
- Desviament total recursos humans = cost total RH estàndard – cost total RH real
- Desviament en una tasca en preu = (cost estàndard – cost real) * hores reals
- Desviament d'un recurs en preu = (cost estàndard – cost real) * consum real
- Desviament en la realització d'una tasca en consum = (consum estàndard – consum real) * cost real
- Desviament total en la realització de tasques = cost total tasca estàndard – cost total tasca real
- Desviament total en recursos = cost total recursos estàndard – cost total recursos real
- Desviament total costos fixos = cost total costos fixos estàndard – cost total costos fixos reals

Mitjançant aquest procés podem observar d'una manera més fàcil les desviacions que hi ha hagut i així poder determinar si és degut a un augment o decrement del nombre d'hores o dels costos.

10. Requisits

En aquesta secció s'especificaran tots els requisits que engloba Hizzy. En primer lloc hi haurà un subapartat per explicar com s'han obtingut els requisits del sistema. En segon lloc una NOT List que permet veure de forma ràpida quines funcionalitats s'inclouen i quines no. A continuació es presenta una llista organitzada per temàtiques dels requisits funcionals. Després es presenta la descripció dels requisits funcionals en forma d'històries d'usuari amb els seus criteris d'acceptació. El següent punt és la descripció dels requisits no funcionals del sistema. Finalment, es documenta el model conceptual de dades.

10.1. Obtenció dels requisits

Com s'ha comentat anteriorment per l'obtenció de requisits s'han realitzat reunions i entrevistes amb les meves companyes de pis i s'ha parlat amb altres persones que es troben en una situació similar. Molts dels requisits s'han proposat per iniciativa personal i després s'ha validat la seva rellevància amb les entrevistes.

Es va fer una reunió inicial explicant els objectius de Hizzy i es va comentar que es volia afegir gamificació a l'aplicació. Les meves companyes van ser qui em van donar la idea de les valoracions i puntuacions pel rànquing. Després els vaig comentar que també estaria bé afegir premis i càstigs per motivar-se més personalment. Sobre els mètodes d'assignació de tasques vaig estar realitzant moltes preguntes en forma d'entrevista de com realment funcionem.

També vaig parlar amb altres persones que comparteixen pis. Em van plantejar l'opció d'assignació de tasques de forma fixa (una persona sempre fa les mateixes tasques). Sorprenentment, vaig trobar força pisos que funcionaven amb aquesta forma d'assignació de tasques i no rotatòriament com fem nosaltres. Per tant, a partir d'aquí vaig decidir afegir els mètodes a l'aplicació.

Més endavant es va fer una reunió amb les companyes per parlar sobre el disseny d'interfície de l'aplicació i de com oferir les funcionalitats que s'havien comentat al principi. Aquesta part es comenta a l'apartat de disseny de la interfície.

Finalment, es va fer una última reunió amb les companyes en la qual es va crear una NOT List inicial.

10.2. NOT List Inicial

La NOT List consisteix en una taula que s'utilitza per definir quines funcionalitats o característiques estaran incloses dins de l'àmbit del projecte i quines quedaran fora.

A la llista es classifica cadascuna d'aquestes funcionalitats en tres grups: **In Scope**, funcionalitats que sí que es trobaran dins del projecte. **Out of Scope**, que són funcionalitats que han estat descartades de l'àmbit del projecte. I finalment, **To Be Determined**, que són aquelles funcionalitats que no hi ha raons per ser descartades, però que tampoc és necessari que estiguin dins l'àmbit del projecte, per tant queden per decidir si s'inclouran o no durant el desenvolupament del projecte.

<table>
<tr><th colspan="2">IN SCOPE</th></tr>
<tr><td>

USERS

1. Gestió d'usuaris

- Login i Logout
- Veure, Editar i Eliminar perfil

2. Gestió de grups

- Veure informació d'un grup
- Crear grup

3. Gestió de mètodes

- Crear i eliminar mètode
- Seleccionar mètode
- Personalitzar mètode

4. Gestió de tasques

- Veure llista de tasques incompletades
- Veure tasques pendents per assignar
- Assignar i Desassignar una tasca
- Crear, Editar i Eliminar tasca
- Completar i Descompletar tasca

5. Valoracions i puntuacions

- Valorar companys
- Veure llista de valoracions pendents
- Veure valoracions rebudes
- Veure rànquing setmanal

</td><td>

6. Premis i penalitzacions

- Escollir premi o penalització
- Veure premis i càstigs pendents
- Acceptar o denegar un premi o càstig

7. Comentaris d'una tasca

- Veure comentaris d'una tasca
- Fer un comentari
- Editar i Esborrar un comentari

8. Usuaris administradors d'un grup

- Editar usuaris del grup
- Editar tasques del mètode
- Canviar mètode del grup
- Editar usuaris del mètode
- Editar usuaris administradors del grup
- Editar i Eliminar grup

SYSTEM

1. Crear tasques obligatòries

2. Control de premis i càstigs

3. Crear rànquing respecte les puntuacions

4. Mitjana de puntuacions d'una tasca

5. Assignar tasques equitativament

</td></tr>
<tr><th>OUT OF SCOPE</th><th>TO BE DETERMINED</th></tr>
<tr><td>

1. Control de despeses comunes

2. Trobar companys de pis

3. Filtrar tasques per nom

</td><td>

1. Checklist en les tasques

2. Rebre notificacions

3. Enllaçar calendari extern

</td></tr>
</table>

Taula 5: NOT List Inicial

10.3. NOT List Final

Finalment, només han de quedar dues columnes **In Scope** i **Out of Scope**, ja que els requisits de **To Be Determined** han de quedar especificats si estan in o out. Per tant, finalment la NOT List queda així:

<table>
<tr><th colspan="2">IN SCOPE</th></tr>
<tr>
<td>

USERS

1. Gestió d'usuaris

- Login i Logout
- Veure, Editar i Eliminar perfil

2. Gestió de grups

- Veure informació d'un grup
- Crear grup

3. Gestió de mètodes

- Crear i eliminar mètode
- Seleccionar mètode
- Personalitzar mètode

4. Gestió de tasques

- Veure llista de tasques incompletades
- Veure tasques pendents per assignar
- Assignar i Desassignar una tasca
- Crear, Editar i Eliminar tasca
- Completar i Descompletar tasca

5. Valoracions i puntuacions

- Valorar companys
- Veure llista de valoracions pendents
- Veure valoracions rebudes
- Veure rànquing setmanal

</td>
<td>

6. Premis i penalitzacions

- Escollir premi o penalització
- Veure premis i càstigs pendents
- Acceptar o denegar un premi o càstig

7. Comentaris d'una tasca

- Veure comentaris d'una tasca
- Fer un comentari
- Editar i Esborrar un comentari

8. Usuaris administradors d'un grup

- Editar usuaris del grup
- Editar tasques del mètode
- Canviar mètode del grup
- Editar usuaris del mètode
- Editar usuaris administradors del grup
- Editar i Eliminar grup

SYSTEM

1. Crear tasques obligatòries
2. Control de premis i càstigs
3. Crear rànquing respecte les puntuacions
4. Mitjana de puntuacions d'una tasca
5. Assignar tasques equitativament

</td>
</tr>
</table>

OUT OF SCOPE
1. Control de despeses comunes 2. Trobar companys de pis 3. Filtrar tasques per nom 4. Checklist en les tasques 5. Rebre notificacions 6. Enllaçar calendari extern

Taula 6: NOT List final

La columna de **In Scope** queda d'igual manera que en la proposta inicial mentre que tota la columna **To Be Determined** passaria a la columna **Out of Scope**. El motiu pel qual s'ha pres aquesta decisió és perquè les tres funcionalitats requerien un temps afegit que no s'havia contemplat i no s'ha arribat a temps per fer-les.

De totes maneres serien unes bones funcionalitats de cara a un futur, ja que és molt útil poder enllaçar les tasques a un calendari extern per combinar-ho amb els altres esdeveniments del dia a dia. A més, rebre notificacions ajudaria a l'usuari a poder-se recordar quan s'ha de fer una tasca. Pel que fa a les *checklist* en les tasques, seria molt útil si es poguessin afegir subtasques indicant petites coses que s'han de fer per completar una tasca.

10.4. Requisits funcionals

Hi ha dos principals actors sobre els requisits funcionals. Aquests dos actors són els usuaris i el sistema.

USUARIS

Els requisits que poden realitzar els usuaris seran totes aquelles funcionalitats que ofereix l'aplicació i que poden controlar els usuaris. Aquests requisits estan dividits per temàtica com es veu a continuació:

1. Gestió d'usuaris

Aquests requisits són tots els relacionats amb la gestió d'usuaris.

- **Login i Logout**

Els usuaris han de poder entrar a l'aplicació amb el seu número de telèfon.
Els usuaris han de poder tancar sessió de forma manual des de l'aplicació.

- **Veure Perfil**

Els usuaris han de poder veure el seu perfil de l'aplicació, és a dir el seu nom i cognoms.

- **Editar Perfil**

Els usuaris han de poder editar el seu perfil de l'aplicació, és a dir el seu nom i cognoms.

- **Eliminar perfil**

Els usuaris han de poder eliminar el seu perfil de l'aplicació. Els canvis que hagin fet aquests usuaris en altres llocs de l'aplicació es mantindran.

2. Gestió de grups

Aquests requisits són tots els relacionats amb la gestió de grups.

- **Veure llista de grups d'un usuari**

Els usuaris han de poder veure la llista dels grups dels quals en són membres. Aquesta llista contindrà informació sobre el nom del grup, el nombre de tasques incompletes en aquell grup en les quals l'usuari n'és assignat, la llista d'aquestes tasques i la posició del rànquing actual.

- **Veure informació d'un grup**

Els usuaris han de poder veure la informació d'un grup del qual en són membres. Aquesta informació inclourà el nom del grup, els noms i cognoms, o número de telèfon si no en tenen, dels membres del grup i el mètode actual seleccionat.

- **Crear grup**

Els usuaris han de poder crear un grup. El grup ha de tenir un nom, uns membres i un mètode. L'usuari que crea el grup es converteix automàticament en usuari administrador d'aquell grup.

3. Gestió de mètodes per assignar tasques

Aquests requisits són tots els relacionats amb la gestió de mètodes.

- **Crear mètode**

Els usuaris han de poder crear un mètode. El mètode ha de tenir un nom, un tipus (rotatori o fix), si el tipus és rotatori, ha de tenir nombre de setmanes de rotació, una llista d'usuaris que formaran part del mètode, una llista de tasques i un grup al qual pertany.

- **Eliminar mètode**

Un usuari ha de poder eliminar un mètode si l'usuari és el creador del mètode o administrador del grup.

- **Seleccionar mètode**

Els usuaris han de poder seleccionar un mètode per a un grup a través d'un desplegable de tots aquells mètodes que pertanyen al grup.

- **Personalitzar mètode**

Els usuaris han de poder personalitzar un mètode canviant els seus atributs: tipus, nombre de setmanes de rotació, nom, la llista d'usuaris i la llista de tasques.

4. Gestió de tasques

Aquests requisits són tots els relacionats amb la gestió de tasques.

- **Veure llista de tasques incompletes**

Els usuaris han de poder veure la llista de tasques en les quals estan assignats però no les han completat encara, per saber quines tasques pendents tenen per fer.

- **Veure tasques pendents per assignar**

Els usuaris han de poder veure una llista de tasques que encara no han estat assignades a ningú, per poder assignar-se-les a ells mateixos i així poder aconseguir punts extra.

- **Assignar i Desassignar una tasca**

Els usuaris han de poder assignar una tasca, si estan creant o editant una tasca i la volen assignar directament a algú o si hi ha una tasca sense assignar, assignar-se-la a ells mateixos.
Els usuaris han de poder desassignar una tasca només si eren els usuaris assignats fins al moment o si són usuaris administradors del grup on pertany la tasca.

- **Veure informació d'una tasca**

Els usuaris han de poder veure la informació d'una tasca. Això inclou el nom, la descripció, el tipus, l'assignat, la data d'inici, la data de fi i si està completada o no.

- **Crear Tasca**

Els usuaris han de poder crear una tasca. La tasca ha de tenir un nom, una descripció, una data d'inici, una data de fi i un grup al qual pertany. A més a més, també podrà tenir un usuari assignat a aquella tasca.

- **Editar Tasca**

Els usuaris han de poder editar una tasca ja creada anteriorment. Per poder-la editar han de ser usuaris administradors del grup on pertany la tasca o els creadors de la tasca o els usuaris assignats.

- **Eliminar Tasca**

Els usuaris han de poder eliminar una tasca ja creada anteriorment. Per poder-la eliminar han de ser usuaris administradors del grup on pertany la tasca o els creadors de la tasca o els usuaris assignats.

- **Completar i Descompletar tasca**

Els usuaris que estan assignats a una tasca han de poder completar-la o descompletar-la per indicar l'estat de la tasca.

5. Valoracions i puntuacions

Aquests requisits són tots els relacionats amb les valoracions entre companys i les puntuacions rebudes en realitzar una tasca, a més a més del rànquing.

- **Valorar companys**

Els usuaris han de poder valorar als companys d'un mateix grup un cop aquests hagin completat una tasca. La valoració anirà del 0 al 5 sent 0 la pitjor nota i 5 la millor nota.

- **Veure llista de valoracions pendents**

Els usuaris han de poder veure una llista amb les valoracions pendents que tenen.

- **Veure valoracions rebudes**

Els usuaris han de poder veure la puntuació final que han obtingut entre la valoració dels seus companys i els punts atribuïts per completar una tasca.

- **Veure rànquing setmanal**

Els usuaris han de poder veure el rànquing setmanal amb la puntuació total de cadascú i la posició en la qual se situen dins del rànquing.

6. Premis i penalitzacions

Aquests requisits són tots els relacionats amb els premis o penalitzacions.

- **Escollir premi o penalització**

Els companys de grup d'un usuari que ha estat premiat o penalitzat han de poder escollir el premi o el càstig corresponent.

- **Veure premis i càstigs pendents**

Un usuari ha de poder veure el nombre de premis i càstigs que té pendents.

- **Acceptar o denegar un premi o càstig**

Un usuari premiat o penalitzat ha de poder acceptar o denegar el premi o càstig que els seus companys proposen.

7. Comentaris d'una tasca

Aquests requisits són tots els relacionats amb els comentaris d'una tasca.

- **Veure comentaris d'una tasca**

Els usuaris han de poder veure els comentaris d'una tasca a la informació de la tasca. Aquests comentaris tindran un autor, una data i un contingut.

- **Fer un comentari**

Els usuaris del grup on pertany la tasca han de poder fer comentaris sobre aquella tasca. El comentari tindrà un contingut.

- **Esborrar un comentari**

Els usuaris creadors d'un comentari han de poder eliminar aquell comentari. Els usuaris administradors del grup on pertany el comentari de la tasca també el podran esborrar.

8. Usuaris administradors d'un grup

Aquests requisits són tots aquells que només poden realitzar els administradors d'un grup.

- **Editar usuaris del grup**

Els usuaris administradors d'un grup han de poder editar els usuaris d'un grup, així com editar-ne els usuaris administradors. Això implica afegir o treure usuaris.

- **Editar tasques del mètode**

Els usuaris administradors d'un grup han de poder editar la llista de tasques d'un mètode. Això implica afegir o treure tasques.

- **Canviar mètode del grup**

Els usuaris administradors d'un grup han de poder canviar el mètode del grup.

- **Editar usuaris del mètode**

Els usuaris administradors d'un grup han de poder editar la llista d'usuaris del mètode. Això implica afegir o treure usuaris.

- **Editar i Eliminar grup**

Els usuaris administradors d'un grup han de poder editar i eliminar el grup del qual en són administradors.

SISTEMA

Les funcionalitats que realitza el sistema són aquelles funcionalitats internes de l'aplicació que no poden ser controlades pels usuaris.

1. Crear tasques obligatòries

El sistema ha de crear les tasques obligatòries de cada grup a l'inici de cada setmana per distribuir-les entre els integrants del grup segons el mètode escollit. Aquestes tasques es crearan internament sense requeriment d'interacció per part de l'usuari.

2. Control de premis i càstigs

El sistema ha de portar un control sobre els premis i els càstigs. Això implica que cada setmana haurà de tenir en compte els rànquings del grup per determinar si hi ha algun integrant del grup que hagi de rebre un premi o un càstig.

3. Crear rànquing respecte a les puntuacions

El sistema haurà de crear un rànquing setmanal cada setmana per cada grup de l'aplicació tenint en compte les puntuacions setmanals de cada integrant del grup.

4. Mitjana de puntuacions d'una tasca

El sistema haurà de portar un control de la mitjana de la puntuació de cada tasca que rep un usuari. Això significa que si el voten dues persones aquell haurà de fer una mitjana entre les dues puntuacions.

5. Assignar tasques equitativament

El sistema haurà d'assignar les tasques de forma la forma més equitativa possible i de forma random el primer cop, després es guiarà a través del mètode. És possible que hi hagi persones descansant alguna setmana depèn del nombre de tasques.

10.4.1. Criteris d'acceptació

Els criteris d'acceptació són els criteris que ha de complir l'aplicació i de quina forma s'ha de comportar perquè es puguin donar a terme les funcionalitats. Aquests criteris han de descriure sempre un context, un esdeveniment i una resposta o conseqüència esperada pel sistema. Per tal de definir-los s'ha utilitzat l'estructura de **Donat-Quan-Aleshores** [35]. A continuació es presenten els criteris d'acceptació de totes les històries d'usuari de l'aplicació. A més, també s'especifica el grau de dificultat de la història d'usuari entre el rang **Alta, Mitjana, Baixa**.

Login i Logout

Com a usuari **vull** accedir o tancar el sistema **per** poder començar a utilitzar l'aplicació o acabar d'utilitzar-la. Aquesta història té complexitat **alta**.

1. **Donada** la pantalla de login de l'aplicació, **quan** l'usuari clica sobre el camp del prefix del telèfon, **aleshores** apareix una pantalla amb una llista de 6 països (Espanya, Anglaterra, Portugal, França, Andorra i Itàlia) en la qual l'usuari pot escollir el seu país del prefix. Aquesta pantalla retornarà a la pantalla de login amb el país seleccionat.

2. **Donada** la pantalla de login de l'aplicació, **quan** l'usuari clica sobre el camp de número de telèfon, **aleshores** apareix el teclat del mòbil de manera que l'usuari pot començar a escriure de forma immediata.

3. **Donada** la pantalla de login de l'aplicació, **quan** els camps prefix del telèfon i número de telèfon han estat omplerts i l'usuari prem el botó *Entrar,* **aleshores** es comproven les credencials al servidor on es troben les dades de tots els usuaris, en cas que siguin correctes s'entra a la pantalla principal o no existeixin, si no és mostra un error i permet a l'usuari modificar les dades.

4. **Donada** la pantalla principal de l'aplicació, **quan** l'usuari obre el menú lateral i prem el botó de *Logout*, **aleshores** es tanca la sessió iniciada i l'usuari torna a la pantalla de login.

Veure Perfil

Com a usuari **vull** accedir al meu perfil **per** poder veure les meves dades en l'aplicació. Aquesta història té complexitat **baixa**.

1. **Donada** la pantalla principal **quan** l'usuari obre el menú lateral, **aleshores** es navega a la pantalla que conté la informació del perfil.

Editar Perfil

Com a usuari **vull** editar el meu perfil **per** poder modificar les meves dades de l'aplicació. Té complexitat **baixa**.

1. **Donada** la pantalla de veure perfil **quan** l'usuari clica sobre els camps de nom i cognom, **aleshores** apareix el teclat del mòbil de manera que l'usuari pot començar a escriure de forma immediata.

2. **Donada** la pantalla de veure perfil **quan** l'usuari ha modificat els camps de nom i/o cognom i prem el botó *Guardar*, **aleshores** es guarden les modificacions al servidor on està la informació de l'usuari.

Eliminar Perfil

Com a usuari **vull** eliminar el meu perfil **per** poder esborrar les meves dades de l'aplicació. Té complexitat **baixa**.

1. **Donada** la pantalla de veure perfil **quan** l'usuari prem la icona de la paperera, **aleshores** apareix una alerta de confirmació per si es vol esborrar aquell compte. En cas afirmatiu s'esborra el compte de l'usuari i es navega a la pantalla de login. En cas negatiu es tanca l'alerta.

Veure llista de grups

Com a usuari **vull** veure la llista de grups dels quals en sóc membre **per** poder saber a quantes i quines tasques em queden per completar en aquell grup, i la posició del rànquing en la que em situo. Té complexitat **alta**.

1. **Donada** la pantalla principal de l'aplicació **quan** l'usuari inicia l'aplicació i ja està loguejat, **aleshores** es carrega una pantalla que conté la llista dels grups dels quals l'usuari n'és membre, amb la seva respectiva informació. Si l'usuari no té cap grup, la llista surt buida.

2. **Donada** qualsevol pantalla de l'aplicació on l'usuari ja estigui loguejat quan l'usuari prem la icona de la *casa* de la barra de navegació, **aleshores** es carrega la pantalla que conté la llista de grups de l'usuari.

Veure informació d'un grup

Com a usuari **vull** veure la informació d'un grup del qual en sóc membre **per** estar informat de totes les característiques del grup, com el nom, els membres i el mètode actiu.

1. **Donada** la pantalla principal amb la llista de grups **quan** l'usuari prem a sobre d'un dels grups, **aleshores** es navega a una pantalla amb tota la informació relativa al grup.

2. **Donada** la pantalla de crear o editar un grup **quan** l'usuari prem el botó *Crear/Guardar* i no hi ha hagut cap error, **aleshores** es navega a la pantalla d'informació d'un grup.

Crear i editar grup

Com a usuari **vull** crear o editar un grup **per** poder utilitzar l'aplicació amb la finalitat esperada. Aquesta finalitat és tenir un lloc comú per poder organitzar-se internament dins d'un mateix grup. La complexitat és **mitjana**.

1. **Donada** la pantalla principal amb la llista de grups **quan** l'usuari clica a la icona *més* o a la icona del *llapis* al costat del nom del grup**, aleshores** es navega a una pantalla amb un formulari per crear o editar un grup.

2. **Donada** la pantalla de d'informació d'un grup **quan** l'usuari prem el botó *Editar*, **aleshores** es navega a una pantalla amb un formulari per crear o editar un grup.

3. **Donada** la pantalla de crear o editar un grup **quan** l'usuari clica sobre el camp nom o número de telèfon, **aleshores** apareix el teclat del mòbil de manera que l'usuari pot començar a escriure de forma immediata.

4. **Donada** la pantalla de crear un grup **quan** l'usuari clica sobre el camp del prefix del telèfon, **aleshores** apareix una pantalla amb una llista de 6 països (Espanya, Anglaterra, Portugal, França, Andorra i Itàlia) en la qual l'usuari pot escollir el país del membre que vol afegir. Aquesta pantalla retornarà a la pantalla de crear un grup amb el país seleccionat.

5. **Donada** la pantalla de crear o editar un grup **quan** els camps prefix del telèfon i número de telèfon estan complets i l'usuari prem el botó *Afegir*, **aleshores** es comproven les credencials del membre que s'ha volgut afegir al grup. Si l'usuari no existeix, es produeix un error mentre que si existeix, l'usuari és afegit a una llista de membres del grup.

6. **Donada** la llista de membres de la pantalla de crear o editar un grup **quan** l'usuari clica la icona de la *creu* al costat del nom del membre, **aleshores** s'elimina l'usuari de la llista de membres del grup.

7. **Donada** la pantalla d'editar grup **quan** es vol canviar de mètode i prem el camp seleccionable *Mètode*, **aleshores** apareix una pantalla amb una llista de tots els mètodes del grup dels quals se'n pot escollir un. Un cop escollit, aquesta pantalla retorna a l'anterior amb un mètode seleccionat.

8. **Donada** la pantalla de crear o editar un grup **quan** estan tots els camps complets i l'usuari prem el botó *Crear/Guardar*, **aleshores** es crea o es modifica un grup correctament i es navega a la pantalla d'informació d'un grup.

Crear i personalitzar un mètode

Com a usuari **vull** crear o editar un mètode **per** poder afegir aquest mètode a un grup i personalitzar-lo al meu gust. La complexitat és **alta**.

1. **Donada** la pantalla de veure informació d'un grup **quan** l'usuari prem a la icona *més*, **aleshores** es navega a una pantalla per crear un mètode que contindrà els camps nom, tipus, setmanes de rotació (si el tipus és rotatori), els membres del mètode i les tasques del mètode.

2. **Donada** la pantalla de veure informació d'un mètode **quan** l'usuari prem el botó *Editar*, **aleshores** es navega a una pantalla editable que conté els camps mencionats anteriorment.

3. **Donada** la pantalla de crear o editar un mètode **quan** l'usuari clica sobre el camp nom o setmanes de rotació, **aleshores** apareix el teclat del mòbil de manera que l'usuari pot començar a escriure de forma immediata.

4. **Donada** la pantalla de crear o editar un mètode **quan** l'usuari clica sobre el camp tipus, **aleshores** apareix una pantalla amb una llista dels dos tipus disponibles (rotatori i fix). Quan s'ha seleccionat un d'aquests dos tipus es torna a la pantalla anterior amb el tipus seleccionat.

5. **Donada** la pantalla de crear o editar un mètode **quan** l'usuari clica sobre el camp membres, **aleshores** apareix una pantalla amb la llista de tots els membres del grup al qual es vol afegir el mètode. Quan s'ha seleccionat el membre que es vol afegir es torna a la pantalla anterior amb el membre seleccionat.

6. **Donada** la pantalla de crear o editar un mètode **quan** l'usuari clica sobre la icona *més* al costat del títol *Tasques Inicials*, **aleshores** es navega a una pantalla per crear una tasca del mètode. Un cop omplerts tots els camps requerits en aquesta pantalla s'afegeix la llista de tasques del mètode.

7. **Donada** la pantalla de crear o editar un mètode **quan** l'usuari clica sobre la icona de la *paperera* al costat del nom de la tasca del mètode, **aleshores** s'elimina el la tasca de les tasques del mètode.

8. **Donada** la pantalla de crear o editar un mètode **quan** l'usuari clica al botó *Crear/Guardar*, **aleshores** es crea o es guarda el mètode i es navega a la pantalla d'informació del mètode.

Eliminar mètode

Com a usuari **vull** eliminar un mètode **per** esborrar-lo de la meva llista de mètodes d'un grup. La complexitat és **baixa**.

1. **Donada** la pantalla de detall del mètode **quan** l'usuari prem la icona de la paperera, **aleshores** s'elimina el mètode del sistema i es navega a la pantalla d'informació del grup.

Seleccionar mètode

Com a usuari administrador **vull** seleccionar un mètode d'assignació de tasques **per** assignar-lo a un grup i indicar-li quines accions ha de realitzar cada setmana i com. La complexitat és **mitjana**.

1. **Donada** la pantalla d'informació d'un grup **quan** l'usuari administrador prem el botó *Canviar Mètode*, **aleshores** es navega a una pantalla amb una llista seleccionable de mètodes del grup. Un cop seleccionat un mètode es torna a la pantalla amb el mètode seleccionat.

2. **Donada** la pantalla d'editar un grup **quan** l'usuari administrador prem sobre el desplegable seleccionable, **aleshores** es navega a una pantalla amb una llista seleccionable de mètodes del grup. Un cop seleccionat un mètode es torna a la pantalla amb el mètode seleccionat.

Veure llista de tasques incompletes

Com a usuari **vull** veure la llista de tasques incompletes a les quals estic assignat **per** poder estar informat de quines tasques em queden pendents per fer. La complexitat és **baixa**.

1. **Donada** la pantalla principal **quan** l'usuari inicia l'aplicació i ja està loguejat, **aleshores** es carrega una pantalla que conté la llista dels grups dels quals l'usuari n'és membre amb la llista de tasques incompletes de l'usuari.

2. **Donada** la pantalla d'informació del grup **quan** l'usuari fa *scroll* fins arribar a l'apartat Tasques Incompletes, **aleshores** es veu la llista de tasques incompletes on l'usuari està assignat.

3. **Donada** la pantalla d'informació del grup **quan** l'usuari prem el botó *Totes*, **aleshores** es navega a una pantalla que conté totes les tasques del grup incompletes de l'usuari i a més les tasques pendents per assignar.

Veure llista de tasques pendents per assignar

Com a usuari **vull** veure la llista de tasques que queden per assignar als grups on són membre **per** poder estar informat de les tasques que em puc assignar per guanyar punts extres. La complexitat és **baixa**.

1. **Donada** qualsevol pantalla on l'usuari estigui loguejat, **quan** l'usuari prem la icona amb l'exclamació de la barra de navegació, **aleshores** es navega a una pantalla on apareix una llista de totes les tasques dels grups de l'usuari que queden pendents per assignar.

2. **Donada** la pantalla d'informació del grup **quan** l'usuari prem el botó *Totes*, **aleshores** es navega a una pantalla que conté les tasques pendents per assignar d'un grup i a més les tasques incompletes del grup de l'usuari.

Assignar tasca

Com a usuari **vull** assignar una tasca a un usuari **per** poder indicar-li que és l'encarregat de realitzar aquella tasca. La complexitat és **baixa**.

1. **Donada** la pantalla de crear o editar una tasca **quan** l'usuari prem sobre el desplegable seleccionable d'*Assignat*, **aleshores** es navega a una pantalla on hi ha una llista de tots els membres del grup. Quan s'ha seleccionat un usuari de la llista es torna a la pantalla anterior amb l'usuari seleccionat.

2. **Donada** la pantalla de la llista de totes les tasques **quan** l'usuari prem el botó *Assignar-me-la*, **aleshores** se li assigna la tasca a ell mateix.

3. **Donada** la pantalla de les tasques que queden pendents per assignar **quan** l'usuari prem el botó *Assignar-me-la*, **aleshores** se li assigna la tasca a ell mateix.

Desassignar tasca

Com a usuari **vull** desassignar una tasca **per** poder deixar-la lliure d'assignats. La complexitat és **baixa**.

1. **Donada** la pantalla de crear o editar una tasca, **quan** l'usuari prem la icona de la *creu* del costat del nom de l'assignat, **aleshores** es desassigna la tasca de l'usuari que hi havia assignat.

Crear i editar una tasca

Com a usuari **vull** crear o editar una tasca **per** poder afegir o modificar tasques dels grups de forma que qualsevol usuari del grup es pugui assignar i realitzar-la. La complexitat és **alta**.

1. **Donada** qualsevol pantalla de l'aplicació on l'usuari estigui loguejat, **quan** l'usuari prem la icona *més* de la barra de navegació, **aleshores** es navega a una pantalla per seleccionar el grup. Un cop seleccionat el grup on es vol afegir la tasca es navega una pantalla amb un formulari amb les propietats que es necessita per crear una tasca. Aquestes propietats són títol, descripció, assignat, data d'inici, tot el dia i data de fi.

2. **Donada** la pantalla d'informació d'una tasca, **quan** l'usuari prem el botó *Edit*, **aleshores** es navega a la pantalla d'editar la tasca.

3. **Donada** la pantalla d'informació d'un grup, **quan** l'usuari prem la icona del *llapis* al costat del nom d'una tasca de la llista de tasques, **aleshores** es navega a la pantalla d'editar la tasca.

4. **Donada** la pantalla de totes les tasques, **quan** l'usuari prem la icona del *llapis* al costat del nom d'una tasca del qual n'és l'assignat, **aleshores** es navega a la pantalla d'editar la tasca.

5. **Donada** la pantalla de crear o editar una tasca, **quan** l'usuari clica sobre el camp títol o descripció, **aleshores** apareix el teclat del mòbil de manera que l'usuari pot començar a escriure de forma immediata.

6. **Donada** la pantalla de crear o editar una tasca, **quan** l'usuari prem sobre el desplegable seleccionable de l'assignat, **aleshores** apareix una pantalla amb la llista de tots els membres del grup. Quan s'ha seleccionat el membre que es vol assignar es torna a la pantalla anterior amb el membre seleccionat.

7. **Donada** la pantalla de crear o editar una tasca, **quan** l'usuari prem un dia del calendari de data d'inici o data fi, **aleshores** se selecciona aquell dia i queda indicat al calendari amb un cercle sobre el dia seleccionat.

8. **Donada** la pantalla de crear o editar una tasca, **quan** l'usuari marca l'opció de *Tot el dia*, **aleshores** el calendari de data fi desapareix, ja que la data fi serà el mateix dia que la data d'inici.

9. **Donada** la pantalla de crear o editar una tasca **quan** l'usuari ha emplenat tots els camps i prem el botó *Crear/Guardar*, **aleshores** es crea o es guarda la tasca en cas que tot estigui

correcte i es navega a la pantalla d'informació de la tasca. Si hi ha algun error, es mostra.

Eliminar tasca

Com a usuari **vull** eliminar una tasca **per** poder esborrar-la de les tasques si tinc la necessitat. Com per exemple si aquella tasca ja no s'ha de fer. La complexitat és **baixa**.

1. **Donada** la pantalla d'informació d'una tasca **quan** l'usuari clica la icona de la *paperera*, **aleshores** s'elimina la tasca i es navega a la pantalla d'informació del grup.

2. **Donada** la pantalla d'informació d'un grup **quan** l'usuari fa *scroll* fins arribar a l'apartat Tasques Incompletes i clica la icona de la *paperera* al costat del nom de la tasca, **aleshores** s'elimina la tasca de la llista.

3. **Donada** la pantalla de totes les tasques **quan** l'usuari prem la icona de la *paperera* al costat del nom d'una tasca, **aleshores** s'elimina la tasca de la llista.

Completar i descompletar tasca

Com a usuari **vull** completar i descompletar una tasca **per** poder indicar quan una tasca ja ha estat realitzada o quan m'he equivocat i la vull descompletar. La complexitat és **baixa**.

1. **Donada** la pantalla principal **quan** l'usuari marca una tasca de la llista de tasques incompletes d'un grup, **aleshores** apareix un botó per completar aquelles tasques seleccionades.

2. **Donada** la pantalla principal **quan** l'usuari prem el botó *Completar Seleccionades*, **aleshores** es completen les tasques seleccionades i desapareixen de la llista.

3. **Donada** la pantalla d'informació d'un grup **quan** l'usuari prem el botó *Completar* d'una tasca, **aleshores** es completa aquella tasca i desapareix de la llista de tasques incompletes.

4. **Donada** la pantalla de totes les tasques **quan** l'usuari prem el botó *Completar* d'una tasca, **aleshores** es completa aquella tasca i desapareix de la llista de tasques incompletes.

5. **Donada** la pantalla d'informació d'una tasca **quan** l'usuari prem el botó *Completar*, **aleshores** es completa aquella tasca.

6. **Donada** la pantalla d'informació d'una tasca **quan** l'usuari prem el botó *Descompletar*, **aleshores** es descompleta aquella tasca.

Valorar companys

Com a usuari **vull** valorar als meus companys **per** poder indicar quina ha estat la meva opinió sobre la realització de la seva tasca. La complexitat és **baixa**.

1. **Donada** qualsevol pantalla de l'aplicació on l'usuari ja estigui loguejat **quan** l'usuari prem la icona de l'estrella de la barra de navegació, **aleshores** es navega a una pantalla que conté totes les valoracions pendents que té l'usuari.

2. **Donada** la pantalla de valoracions pendents **quan** l'usuari prem sobre d'una de les 5 estrelles que té cada tasca i prem al botó *Valorar*, **aleshores** es processa la valoració i aquella tasca desapareix de la llista de tasques pendents per valorar.

Veure llista de valoracions pendents

Com a usuari **vull** veure la llista de valoracions pendents que tinc **per** estar informat de les valoracions que em queden per fer i així fer-les. La complexitat és **mitjana**.

1. **Donada** qualsevol pantalla de l'aplicació on l'usuari ja estigui loguejat **quan** l'usuari prem la icona de l'*estrella* de la barra de navegació, **aleshores** es navega a una pantalla on hi ha una llista de les valoracions pendents de l'usuari.

Veure valoracions rebudes

Com a usuari **vull** veure la llista de valoracions que he rebut **per** saber quina és la nota mitjana que he rebut sobre la tasca que he fet. La complexitat és **baixa**.

1. **Donada** qualsevol pantalla de l'aplicació on l'usuari ja estigui loguejat **quan** l'usuari prem la icona de l'*estrella* de la barra de navegació, **aleshores** es navega a una pantalla on hi ha una llista de les valoracions rebudes de l'usuari.

Veure rànquing setmanal

Com a usuari **vull** poder veure el rànquing setmanal de cada grup del qual en sóc membre **per** poder saber en quina posició em situo i en quina se situen els meus companys. La complexitat és **mitjana**.

1. **Donada** qualsevol pantalla de l'aplicació on l'usuari ja estigui loguejat **quan** l'usuari prem la icona del pòdium de la barra de navegació, **aleshores** es navega a una pantalla que conté el rànquing setmanal dels grups als quals pertany l'usuari.

Escollir premi o penalització

Com a usuari premiat o com a company d'un usuari castigat **vull** poder escollir el premi o el càstig que jo vulgui **per** rebre el premi o castigar al meu company. La complexitat és **baixa**.

1. **Donada** la pantalla principal de l'aplicació **quan** l'usuari prem la icona per obrir el menú lateral i prem a *Bonificacions*, **aleshores** es navega a una pantalla on hi ha els premis i càstigs pendents, si n'hi ha algun, llavors hi ha un botó d'escollir-lo.

Veure premis i càstigs pendents

Com a usuari **vull** poder veure els premis i càstigs que tinc pendents **per** saber quants són i quins són. La complexitat és **alta**.

1. **Donada** la pantalla principal de l'aplicació **quan** l'usuari prem la icona per obrir el menú lateral i prem a *Bonificacions*, **aleshores** es navega a una pantalla on hi ha els premis i càstigs pendents.

Acceptar o denegar un premi o càstig

Com a usuari **vull** poder acceptar o denegar un càstig **per** si no m'agrada o m'agrada i així poder rebre'l o donar-lo. La complexitat és **baixa**.

1. **Donada** la pantalla principal de l'aplicació **quan** l'usuari prem la icona per obrir el menú lateral i prem a *Bonificacions*, **aleshores** es navega a una pantalla on hi ha els premis i càstigs pendents per acceptar.

2. **Donada** la pantalla de premis i càstigs **quan** l'usuari prem el botó *Acceptar*, **aleshores** s'accepta el premi o càstig.

3. **Donada** la pantalla de premis i càstigs **quan** l'usuari prem el botó *Denegar,* ***aleshores*** es denega el premi o càstig.

Veure comentaris d'una tasca

Com a usuari **vull** poder veure els comentaris d'una tasca **per** estar informat del que s'ha comentat d'aquella tasca entre els companys del grup. La complexitat és **mitjana**.

1. **Donada** la pantalla d'informació d'una tasca **quan** l'usuari fa *scroll* fins arribar a l'apartat de comentaris, **aleshores** es mostra una llista amb tots els comentaris que s'ha fet d'aquella tasca.

Fer un comentari a una tasca

Com a usuari **vull** poder comentar una tasca **per** poder afegir informació addicional sobre la tasca. La complexitat és **baixa**.

1. **Donada** la pantalla d'informació d'una tasca **quan** l'usuari prem el botó *Comentar*, **aleshores** es navega a una pantalla on hi ha un camp de text per emplenar que serà el contingut del comentari.

2. **Donada** la pantalla de crear un comentari **quan** l'usuari clica sobre el camp contingut, **aleshores** apareix el teclat del mòbil que permet a l'usuari escriure immediatament.

3. **Donada** la pantalla de crear un comentari **quan** ja hi ha el contingut omplert i l'usuari prem el botó *Comentar*, **aleshores** es crea el comentari i es navega a la pantalla d'informació d'una tasca.

Esborrar un comentari

Com a usuari **vull** poder esborrar un comentari d'una tasca **per** poder eliminar aquella informació de la tasca. La complexitat és **baixa**.

1. **Donada** la pantalla d'informació d'una tasca **quan** l'usuari prem la icona de la *paperera* al costat del comentari, **aleshores** s'esborra el comentari de la llista de comentaris de la tasca.

10.5. Requisits no funcionals

L'analisi de requisits no funcionals s'ha dut a terme seguint les pautes marcades pel document "Volere Requirements Specification Template" [12]. Tot i que s'han analitzat totes les categories especificades al document, a continuació només s'especificaran les categories de les quals aquest projecte té requisits rellevants.

10. Look and feel

10a. Appearance Requirements

- L'aparença de l'aplicació ha de ser atractiva pels usuaris que l'utilitzin.

11. Usability and Humanity Requirements

11a. Ease of Use Requirements

- L'aplicació ha de ser fàcil d'utilitzar i intuïtiva.

- El producte ha d'ajudar als usuaris a no cometre errors.

11b. Personalization and Iternationalization Requirements

- L'aplicació serà multiidioma. Els idiomes disponibles són anglès, català i castellà.

10.6. Model conceptual de dades

En aquest apartat s'explicarà el disseny conceptual de la Base de Dades que l'aplicació utilitzarà per emmagatzemar tota la informació necessària.

Per poder crear aquesta base de dades s'hauran d'identificar primer tots els objectes implicats en l'aplicació.

- **Usuaris**: són els usuaris registrats de l'aplicació.
- **Grups**: són els grups d'usuaris que té l'aplicació. Com per exemple el pis on vius.
- **Tasques**: són les tasques que s'han de realitzar pels usuaris que pertanyen a un grup. Com per exemple netejar la cuina.
- **Mètodes**: són els mètodes que pot tenir un grup.
- **Comentaris**: són els comentaris de les tasques.
- **Bonificacions**: són els premis i els càstigs que s'hauran d'acceptar un cop es proposin pels companys de grup.
- **Rànquings**: són els rànquings setmanals de cada grup.

El següent diagrama UML sobre la Base de Dades mostra la relació entre els objectes i a quines restriccions adicionals estan sotmesos:

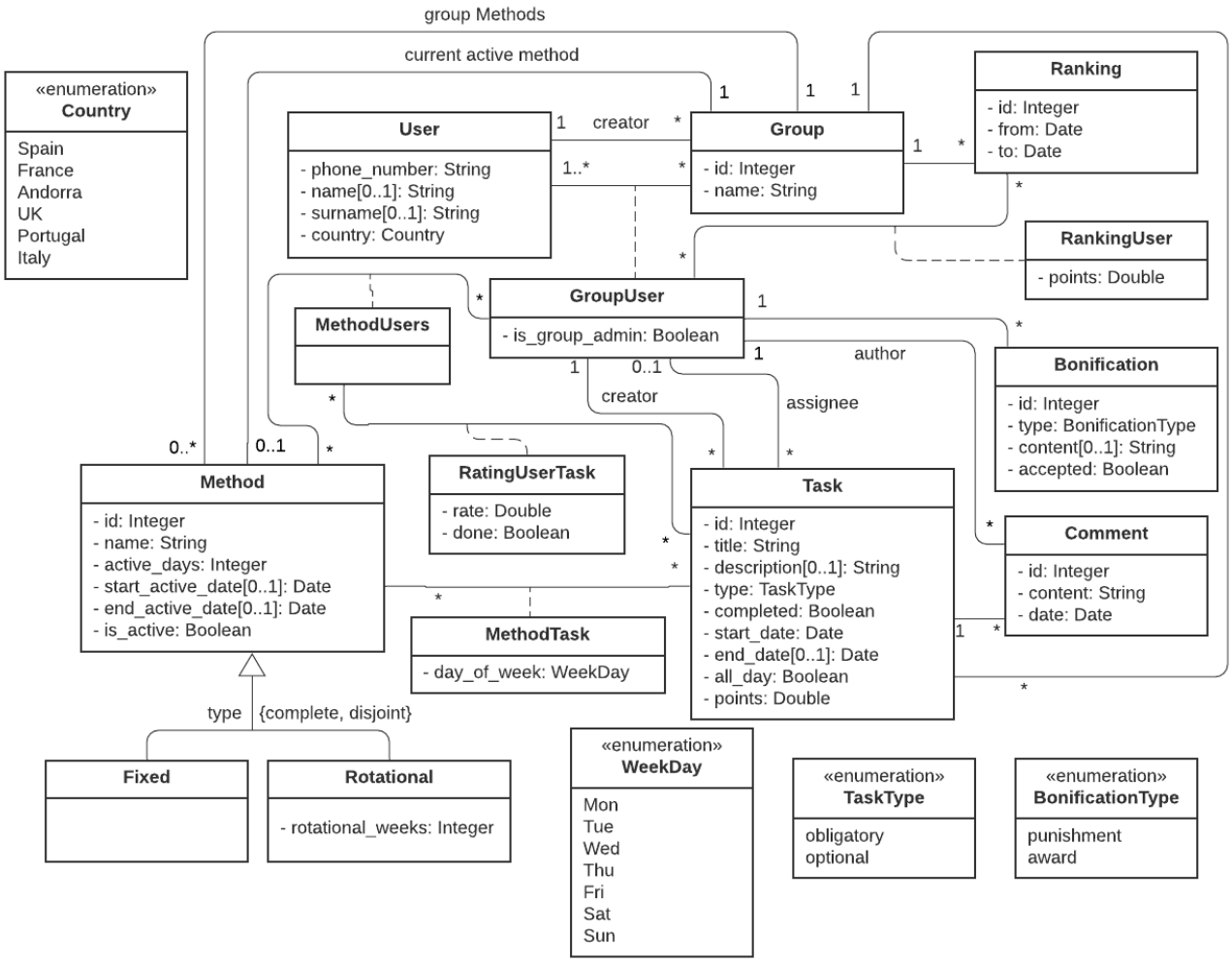


Figura 3: Model conceptual de dades

Restriccions:

- **Claus Externes**: (User, phone_number); (Group, id); (Bonification, id); (Comment, id); (Task, id); (Method, id), (Ranking, id)

Restriccions textuals:

- **(RT1)** El GroupUser d'un Comment ha de ser del mateix Group al qual pertany la seva Task
- **(RT2)** Els GroupUsers d'un Method han de ser del mateix Group que el Method
- **(RT3)** Els points d'una Task només seran més grans que 0 si completed és true
- **(RT4)** El boolean is_active de Method només serà true si el Method és el current active method del Group
- **(RT5)** El creador d'un Group serà automàticament un GroupUser d'aquell mateix Group amb is_group_admin amb valor true

- **(RT6)** Una Task només pot pertànyer a un Method si el Group del Method és el mateix que el Group de la Task
- **(RT7)** Un GroupUser assignee d'una Task només pot ser-ho si és usuari del mateix Group al qual pertany aquella Task
- **(RT8)** L'author d'un Comment d'una Task només pot ser-ne l'autor si pertany al mateix Group que el Group de la Task
- **(RT9)** El creator d'una Task només pot ser-ne creador si pertany al mateix Group que el Group de la Task
- **(RT10)** Els GroupUsers d'un Ranking només podran formar part del Ranking si pertanyen al mateix Group on pertany el Ranking
- **(RT11)** Els RatingUsersTask d'una Tasca i un GroupUser han de pertànyer al mateix Group

11. Arquitectura

11.1. Visió general

En primer lloc cal parlar dels diferents elements que formen part del sistema i com estan relacionats.

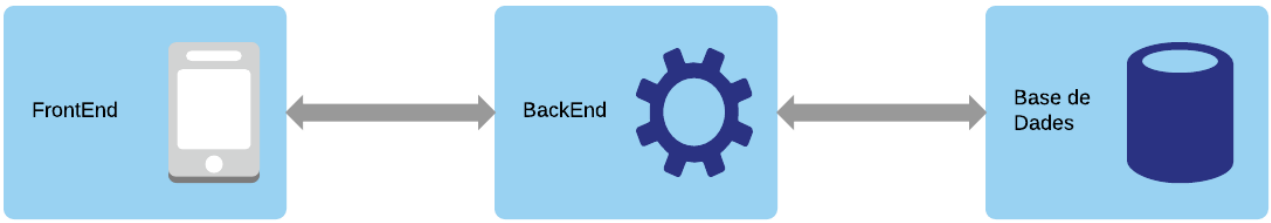


Figura 4: Diagrama de l'arquitectura

Al diagrama anterior es pot veure que al sistema hi ha 3 elements principals:

- **Frontend** és la part que conté la interfície d'usuari, que es comunica amb el backend.
- **Backend** és l'element que conté la lògica de l'aplicació i que rep les peticions del frontend a partir de les quals es comunica amb la Base de Dades per realitzar els canvis necessaris.
- La **Base de Dades** és on s'emmagatzema tota la informació sobre els mapejos del sistema i les dades dels usuaris.

11.2. Frontend

El frontend està dissenyat amb el patró Model-Vista-Controlador (MVC). En l'arquitectura MVC, la Vista és la interfície d'usuari. El Model és la lògica de domini, que és l'element que s'encarrega de fer peticions al backend, que és on es troba implementada la lògica. I finalment, el Controlador és la part de l'arquitectura que fa de pont entre la Vista i el Model.

Per tant, l'arquitectura de frontend quedaria representada de la següent manera:

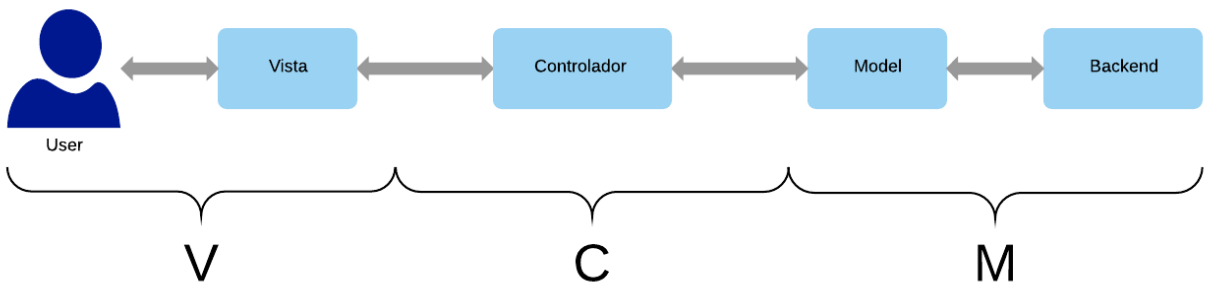


Figura 5: Arquitectura del frontend

Tal com es pot veure a la Figura 4, l'usuari interactua amb la Vista, la qual rep les peticions de realitzades per l'usuari. La Vista transmet aquestes peticions al Controlador, que les deriva al Model i que aquest finalment les envia al Backend de l'aplicació.

Un cop el Backend ha processat la petició, envia la resposta al Model, qui la retorna al Controlador que la processa per tal de poder-la mostrar a la Vista.

11.3. Backend

El Backend de l'aplicació està estructurant en forma d'API REST, que es comunicarà a través de fitxers JSON. JSON (JavaScript Object Notation) és un format per guardar i intercanviar dades que és sintàcticament idèntic al codi per crear objectes JavaScript. D'aquesta forma, un programa JavaScript pot convertir molt fàcilment dades JSON en objectes natius de JavaScript.

El format JSON està escrit en parelles clau/valor.

```
myObj = { "name":"John", "age":30, "car":null };
x = myObj["name"];
```

Figura 6: Exemple format JSON

On per exemple la clau seria name, i John en seria el valor. Per accedir als valors es pot fer referenciant a la clau com en l'exemple.

El fet que sigui una API REST implica que l'estructura i funcionament del backend tindrà les següents característiques:

- L'API no guardarà informació sobre l'estat, per tant tota la informació sobre la sessió es guardarà a la banda del client.

- L'accés als diferents recursos es donarà a través de URLs que identificaran cadascun d'aquests recursos. L'estructura de les URLs és la següent: **"protocol"://"host":"port"/ "endpoint"?"paràmetres per filtrar"**

- Els mètodes que s'utilitzaran per interactuar amb els diferents recursos són els mètodes: **GET** - per llegir, **POST** - per crear, **PUT** - per editar, **DELETE** - per eliminar.

- Les respostes es faran amb *HTTP Response Status Codes* [18], els estats que s'utilitzaran a l'aplicació s'especifiquen a la següent taula.

Grup	Estat	Missatge
Successful Responses **2xx**	200	OK
	201	Created
	204	No Content
Client Error Responses **4xx**	400	Bad Request
	401	Unauthorized
	403	Forbidden
	404	Not Found
Server Error Responses **5xx**	500	Internal Server Error

Taula 7: HTTP Response Status Codes

Els avantatges d'utilitzar una API REST és que et permet utilitzar els protocols que ja existeixen, això implica que els desenvolupadors no han d'instal·lar software adicional quan creen una API REST. A més proporciona una gran flexibilitat, això implica que poden manegar diferents tipus de crides i retornar les dades en diferents formats.

L'arquitectura del backend quedarà de la seguent forma:

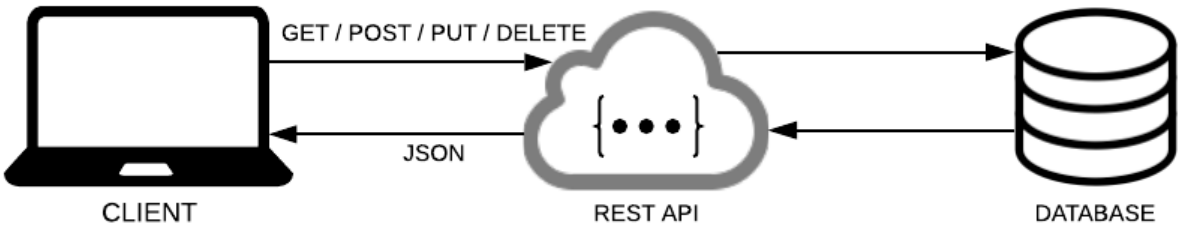


Figura 6: Arquitectura Barckend

El client envia la petició a l'API REST, que s'encarrega d'executar la lògica de la petició i accedeix a la base de dades per realitzar les *queries* i canvis necessaris.

Finalment, quan rep una resposta l'envia en format JSON cap al client.

11.4. Diagrama de classes de disseny

En aquest apartat s'explicarà com s'ha distribuït la base de dades. Tenint en compte el model conceptual de dades s'ha dissenyat el diagrama de classes on bàsicament s'eliminen les classes associatives. El diagrama quedarà així:

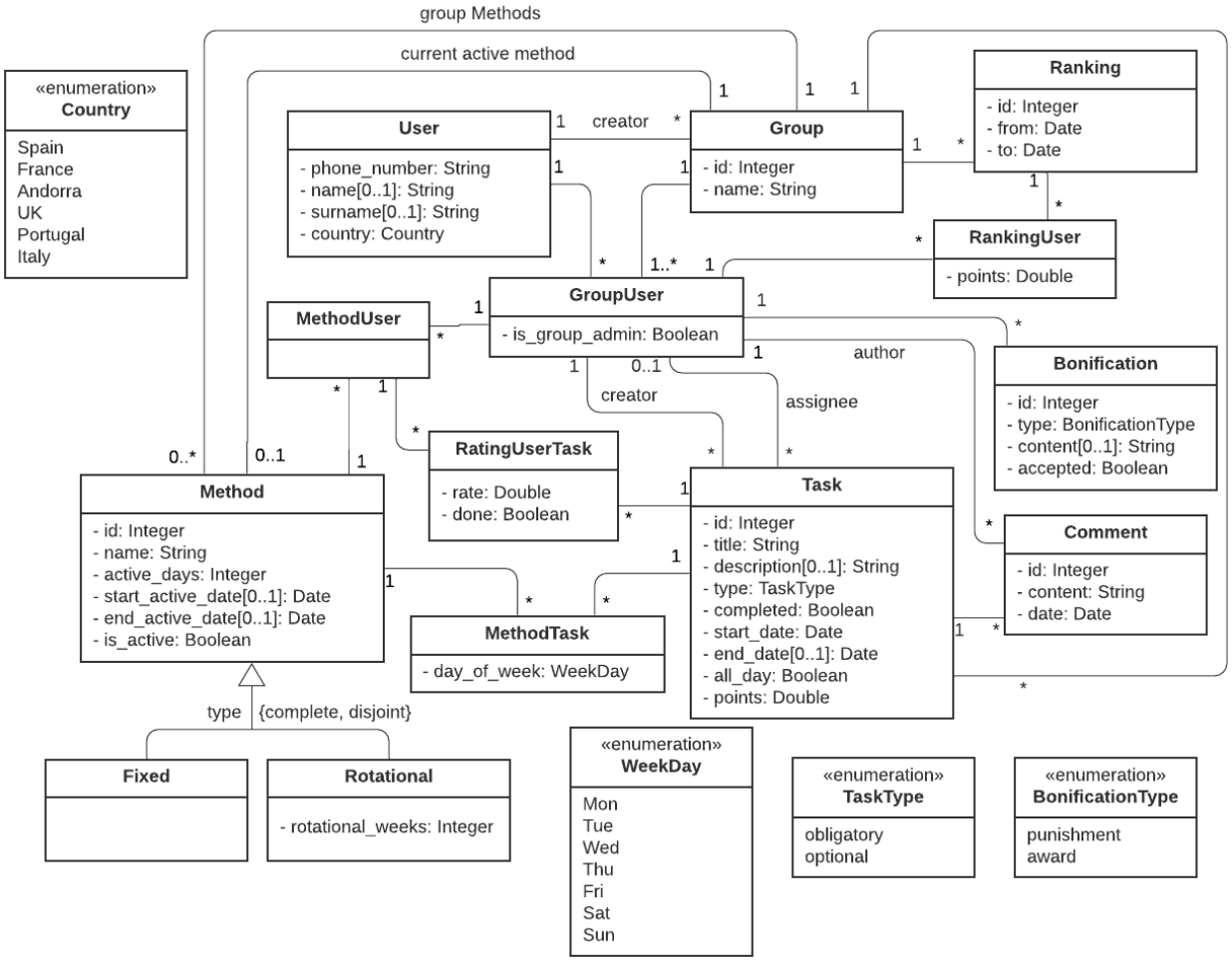


Figura 7: Diagrama de classes

A l'eliminar les classes associatives es crearan les següents identitats a més de les anteriors:

- **GroupUser**: són els usuaris que pertanyen a un grup.
- **MethodTask**: són les tasques inicials que pertanyen a un mètode.
- **MethodUser**: són els usuaris que formen part del mètode.
- **RankingUser**: són els usuaris que pertanyen a un rànquing.
- **RatingUserTask**: són els usuaris del mètode que valoren una tasca.

A més, s'afegeixen les següents restriccions a les anteriors:

Restriccions:

- **Claus Externes**: (GroupUser, User::phone_number, Group::id); (MethodTask,Task::id, Method::id); (MethodUser, User::phone_number, Group::id, Method::id); (RankingUser, Ranking::id, Group::id, User::phone_number); (RatingUserTask, Group::id, User::phone_number, Method::id, Task::id)

A partir de tot això, per la base de dades s'han creat les taules corresponents al diagrama de classes i l'estratègia que s'ha fet servir per les subclasses és **Single Table Inheritance**. Aquesta estratègia consisteix en utilitzar una única taula per tots els mètodes afegint un camp a la taula que s'anomena type, que indicarà de quin tipus és. Finalment la declaració de la taula quedaria així:

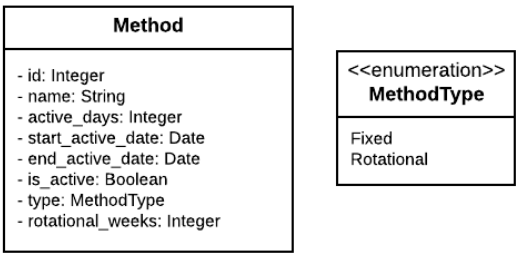


Figura 8: Gerarquia de classes amb Single Table Inheritance

11.5. Interfície

A continuació es mostraran els dissenys d'algunes de les pantalles de l'aplicació, tots els Mockups han estat creats mitjançant l'eina Figma.

Pantalla Login

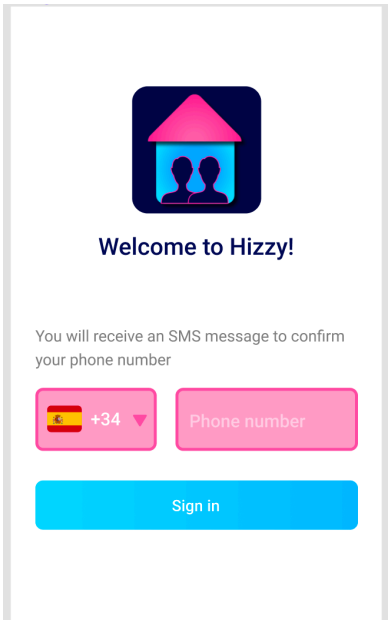


Figura 9: Mockup Pantalla Login

Aquesta pantalla serà la primera que veuran els usuaris que no estiguin ja loguejats dins l'aplicació. La imatge a la part superior de la pantalla és el Logo de l'aplicació. Aquest Logo és una casa ja que fa referència al nom de l'aplicació, Hizzy. Dins la casa hi ha dues persones per simbolitzar que aquesta aplicació fa referència a la convivència entre persones. A la part inferior hi haurà un desplegable per poder escollir el prefix del número de telèfon que s'ha d'introduir, el qual servirà per identificar a l'usuari. Al costat del desplegable hi haurà un camp de text per introduir el número de telèfon i un botó per confirmar.

Pantalla principal

Figura 10: Mockup Pantalla Principal

Aquesta pantalla serà la primera que veuran els usuaris un cop ja estiguin loguejats a l'aplicació. Tindrà una capçalera amb el nom de l'aplicació en la qual hi ha dues icones una a cada costat. La icona de l'esquerra serà un menú lateral en el qual hi haurà algunes funcionalitats com per exemple fer Logout. La icona de la dreta serà per afegir un nou grup.

Pel que fa a la informació proporcionada per la pantalla, es podrà veure una llista dels grups de l'usuari amb alguna informació com el nom, les tasques en les quals l'usuari està assignat i estan pendents per completar, la posició actual en el rànquing setmanal i la contribució setmanal en sobre el total de tasques del grup. A més, les tasques incompletes es podran completar des d'aquesta pantalla. Per cada grup si es prem dins la zona de la vista que ho engloba, navegarà a una pantalla que tingui tota la informació del grup. Si es premen les dues icones de la dreta del nom (el llapis i la paperera), es podrà editar o esborrar el grup.

A la part inferior de la pantalla hi haurà una barra de navegació amb les funcionalitats més importants. Estarà disponible a totes les pantalles de l'aplicació excepte a la pantalla de Login.

La idea de la barra de navegació va aparèixer durant una reunió amb les companyes de pis. Elles em van donar la idea d'afegir una barra de navegació amb unes quantes funcionalitats com té Instagram. Vam estar debatent quines serien les més importants de posar. Així doncs, les 5 icones de la barra de navegació representaran:

- **Casa**: Indicarà la pàgina principal.
- **Estrella**: Navegarà a la pantalla de valoracions.
- **Símbol de suma**: Navegarà a la pantalla de nova tasca.
- **Símbol amb exclamació**: Navegarà a la pantalla de tasques pendents per assignar, és a dir aquelles que encara no tinguin un assignee.
- **Barres**: Navegarà a la llista de rànquings setmanals.

Pantalla Informació del Grup

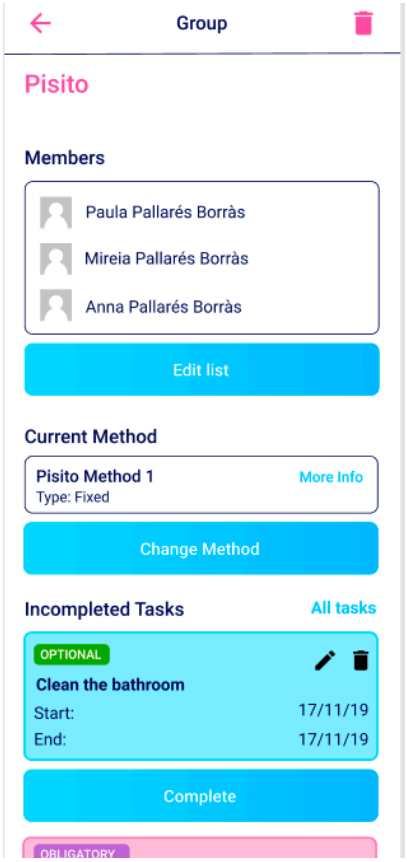


Figura 11: Mockup Pantalla Informació del Grup

Aquesta pantalla contindrà tota la informació d'un grup. A la part superior de la pantalla hi haurà una capçalera indicant que es tracta de grups, amb dues icones, una de tirar enrere i l'altra per eliminar el grup.

A la part central hi haurà la informació pertinent, com el nom del grup, la llista de membres (la qual es podrà editar), el mètode actual actiu (el qual es podrà canviar o veure informació amb més detall) i una llista de tasques incompletes, en les quals l'usuari n'és l'assignee, amb un resum de la seva informació. Aquestes tasques es podran completar des d'aquesta pantalla. A més també es podrà editar i eliminar una tasca. Si es prem a la vista que engloba una tasca es navegarà a una pantalla amb informació més detallada de la tasca en concret.

Finalment, si es prem el text de *All tasks* es navegarà a una pantalla amb totes les tasques incompletes del grup, no només les quals l'usuari n'és l'assignee.

Pantalla Editar Grup

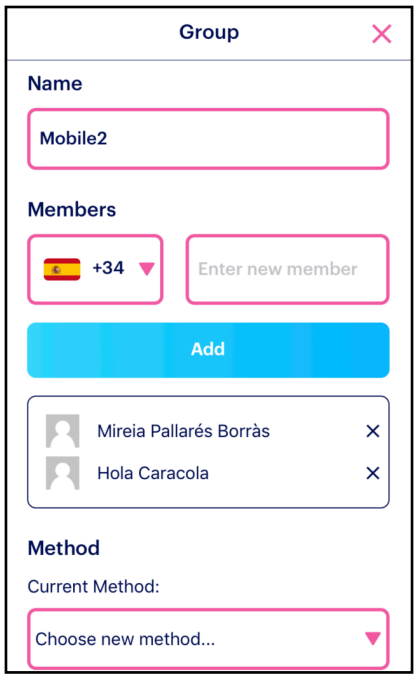


Figura 12: Mockup Editar Grup

Aquesta pantalla contindrà un formulari amb la informació del grup que l'usuari haurà de proporcionar. Tindrem el nom del grup, els membres i el mètode actual.

Per als membres s'afegiran de la mateixa manera que al Login, amb el desplegable del prefix telefònic i el TextInput per escriure el número de telèfon. A més també tindrem la llista de membres actual en la qual es podran eliminar també els usuaris que hi ha a la llista.

Per al mètode hi haurà un desplegable amb tots els mètodes del grup per poder escollir el que sigui més adequat per a l'usuari en aquell moment.

Pantalla Editar Mètode

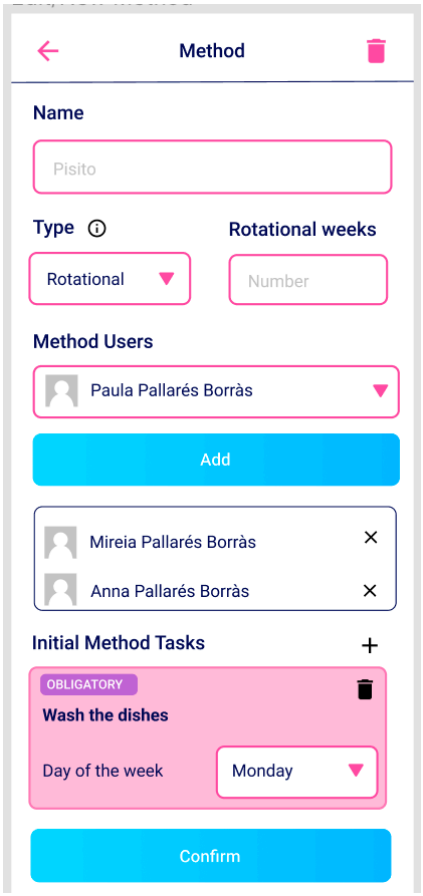


Figura 13: Mockup Editar Mètode

Aquesta pantalla contindrà un formulari amb la informació del mètode que l'usuari haurà de proporcionar. Tindrem el nom del mètode, el tipus, els rotational weeks en cas que el tipus sigui rotational, els method users i les initial method tasks.

Per al tipus hi haurà un desplegable amb les dues possibles opcions: fix i rotatori. La icona que hi ha al costat del type navegarà a una pantalla on s'expliquin els dos tipus de mètodes.

Per als usuaris del mètode hi haurà un desplegable amb tots els usuaris del grup al qual pertany el mètode i un botó per afegir a aquell usuari seleccionat. A més també hi haurà una llista amb els usuaris que ja pertanyen al mètode.

Per les tasques inicials del mètode hi haurà la llista de les tasques amb el dia de la setmana que s'hauran de començar. Aquest desplegable contindrà tots els dies de la setmana. Per afegir una nova tasca al mètode s'haurà de clicar al botó + i si es vol eliminar una que ja n'és part s'haurà de

prémer a la paperera. Si es prem a la vista que engloba la tasca, es navegarà a una pantalla amb informació més detallada sobre la tasca.

Finalment, hi ha un botó per confirmar els canvis.

Pantalla Valoracions

Figura 14: Mockup Pantalla Valoracions

Aquesta pantalla estarà dividida en dos tipus de llistes. La primera serà les valoracions pendents que té un usuari. Un cop s'hagin decidit els punts que es donaran al company, apareixerà un botó per enviar la valoració.

La segona llista seran les valoracions mitjanes rebudes per les tasques que ja ha realitzat un usuari. Aquestes valoracions seran anònimes per evitar problemes i confusions.

En qualsevol de les dues llistes si es prem la vista que engloba qualsevol tasca, es navegarà a una pantalla amb més informació detallada sobre la tasca.

11.6. Diagrama de navegació

Un diagrama de navegació és un document que s'utilitza per explicar l'estructura de la navegació, en aquest cas la navegació entre les pantalles de Hizzy. Per entendre primer la navegació s'han d'explicar breument les pantalles que formen part de l'aplicació. Aquestes pantalles són:

- **Login**. És la pantalla per iniciar sessió a l'aplicació.
- **Main**. És la pantalla principal de l'aplicació on es poden fer moltes funcionalitats i conté la llista de grups de l'usuari.
- **Account**. És la pantalla del perfil de l'usuari.
- **NewEditTask**. És la pantalla de crear o editar una tasca.
- **NewEditGroup**. És la pantalla de crear o editar un grup.
- **GroupDetail**. És la pantalla on hi ha la informació d'un grup.
- **TaskDetail**. És la pantalla on hi ha la informació d'una tasca.
- **NewEditMethod**. És la pantalla de crear o editar un mètode.
- **MethodDetail**. És la pantalla on hi ha la informació d'un mètode.
- **Rankings**. És la pantalla on es poden veure els rànquings setmanals de cada grup.
- **PendingTasks**. És la pantalla on es poden veure aquelles tasques pendents per assignar.
- **AllTasks**. És la pantalla on es poden veure les tasques incompletades d'un grup i les que queden pendents per assignar.
- **Ratings**. És la pantalla on es poden veure les valocacions rebudes i les valoracions pendents per fer i valorar.
- **SelectGroup**. És la pantalla on es selecciona a quin grup vols crear una tasca.
- **SelectCountryCode**. És la pantalla on es selecciona el prefix del telèfon de l'usuari.
- **SelectMethodType**. És la pantalla on es selecciona el tipus de mètode.

- **SelectMember**. És la pantalla on es selecciona un membre del grup.
- **SelectWeekDay**. És la pantalla on es selecciona el dia de la setmana.
- **SelectMethod**. És la pantalla on es selecciona un mètode entre tots els mètodes d'un grup.
- **Bonifications**. És la pantalla on es poden veure les bonificacions pendents.
- **ChooseBonification**. És la pantalla on es pot escollir la bonificació.
- **NewComment**. És la pantalla per crear un comentari.

A partir d'aquí, la navegació quedaria de la següent forma:

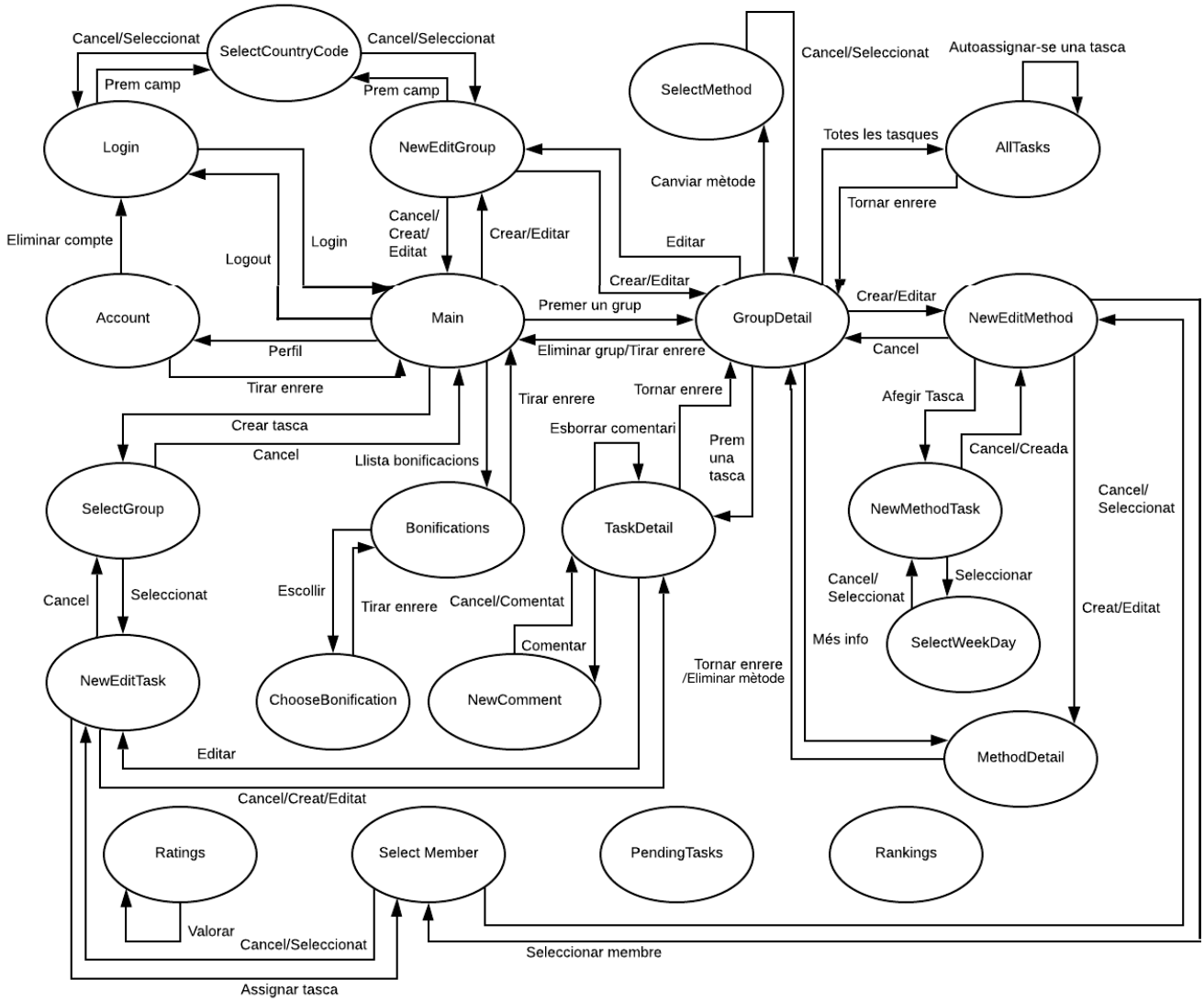


Figura 15: Diagrama de navegació de l'aplicació

S'ha de tenir en compte que des de qualsevol pantalla excepte la de Login i la de SelectCountryCode, es pot navegar a les cinc pantalles de la barra de navegació (Main, Ratings, Rankings, SelectGroup i PendingTasks). Aquestes fletxes no s'han afegit per no embolicar més el diagrama.

11.7. Patrons

Per desenvolupar l'aplicació s'han anat necessitant alguns patrons:

Observer

El patró observador s'utilitza a la llibreria Mobx [20] utilitzada per la part de frontend, explicada més endavant. En aquest patró hi ha dos tipus de declaracions, els components o classes **@observer**, que són els que s'encarreguen d'observar qualsevol canvi que detectin en aquells altres components que són **@observable**, que són components o propietats que deixen ser observats pels observers. Quan un observer detecta un canvi en algun component o propietat observable, realitza els canvis necessaris.

Si per exemple tenim una classe @observer amb variable de classe @observable:

```
@observer
class BaseMainScreen extends Component {
  static propTypes = {
    navigation: PropTypes.object,
    screenProps: PropTypes.object
  };

  colors = [
    { background: "#34D6FE90", border: "#34D6FE" },
    { background: "#FF83B590", border: "#FF83B5" },
    { background: "#000E5440", border: "#000E5490" }
  ];

  @observable
  userGroups = [];
```

Figura 16: Exemple codi Patró Observer

Si hi ha algun canvi en la variable userGroups la classe ho detecta i torna a refrescar les dades, és a dir, si la classe és una pantalla, tornarà a renderitzar els components de la pantalla amb les noves dades en *run time*. En canvi, si hi hagués algun canvi en la variable colors, la classe no es tornaria a renderitzar i per tant, encara veuríem els colors antics perquè no es canviarien en *run time*. S'hauria de refrescar manualment.

Decorator

El patró decorador s'utilitza en forma de HOC, Higher Order Component [21]. El HOC s'utilitza per reaprofitar lògica del component. És a dir, si hi ha per exemple dos components que sempre

utilitzen les mateixes funcions, aquestes funcions se situaran al HOC en lloc de als dos components perquè no es repeteixi codi innecessari. Per tant cada cop que s'utilitzi un d'aquests components s'haurà de cridar primer al decorador per afegir aquestes funcions necessaries.

12. Implementació

En aquesta secció es detallarà el procés d'implementació de l'aplicació.

12.1. Backend

12.1.1. Esquema general

El llenguatge de programació escollit per desenvolupar l'API de l'aplicació és Django, a través del framework Django REST Framework [22]. La següent imatge representa un esquema per entendre millor com funciona aquest *framework*:

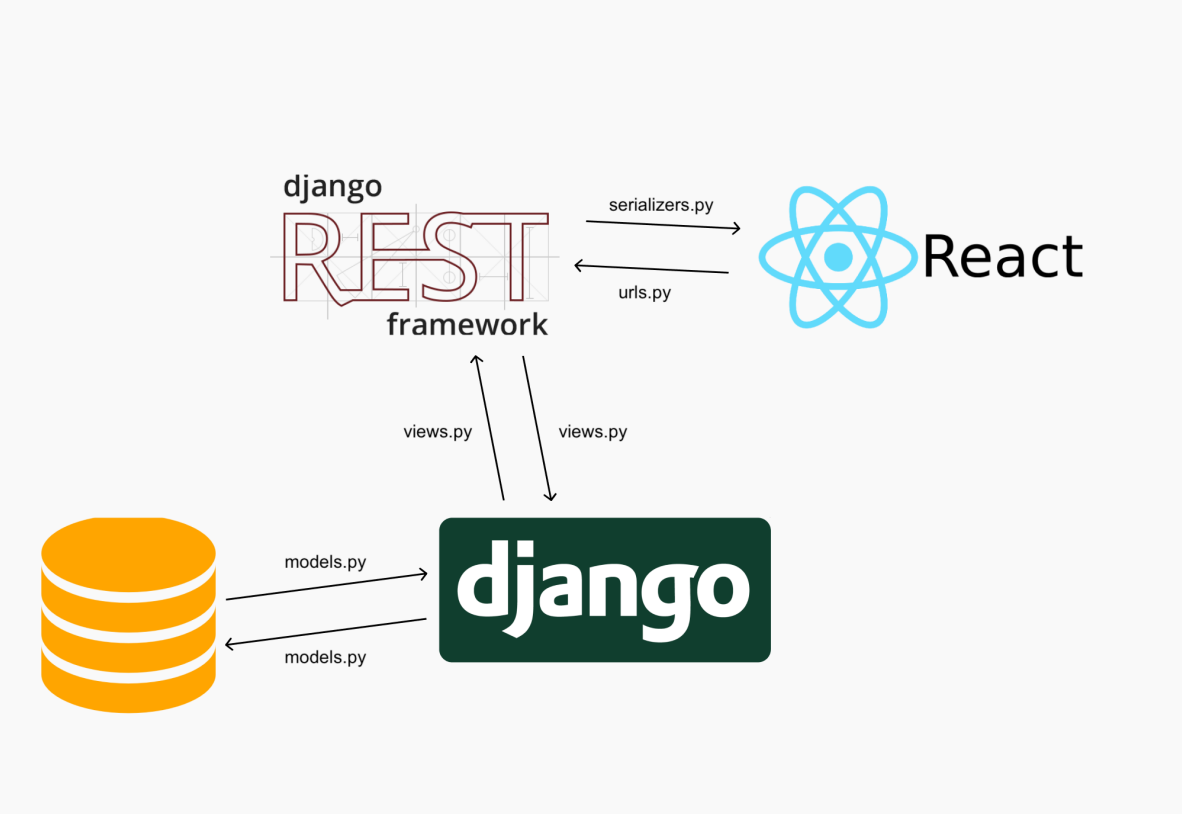


Figura 17: Esquema funcionament Django REST

El principal motiu pel qual he escollit Django és perquè Django per defecte ja té instal·lada una base de dades SQLite [23]. SQLite és una llibreria que implementa base de dades SQL transaccional. S'ha decidit utilitzar aquesta base de dades, ja que, com l'*scope* del projecte no era tan gran, ja era una base de dades adequada per al seu funcionament. Utilitzar una base de dades relacional implica una serie d'avantatges:

- **Atomicitat**. Significa que si hi ha alguna fallada durant la transacció, el codi no s'executarà per a res, en canvi si no hi ha cap fallada s'executara al 100%.

- **Estàndards ben definits**. Significa que les dades sempre estan estructurades de la mateixa manera i la manera de fer les transaccions sempre és la mateixa i està ben definida.

El segon motiu pel qual he escollit Django és perquè Django ofereix un ORM, Object Relational Mapping. ORM és una tècnica d'accés a una base de dades relacional que utilitza el paradigma *object-oriented*. És el mapeig de les dades de la base de dades en forma de funcions sense la necessitat d'escriure crides SQL. Per exemple:

```
queryset = GroupUser.objects.all()
```

Figura 18: Exemple d'ORM en Django

Aquesta funció retornarà totes les *entries* de la taula de la GroupUser, sense la necessitat d'escriure codi SQL.

Per connectar-se amb la base de dades Django utilitza el fitxer **models.py.** A través de fer migracions amb una comanda ORM, actualitza l'estructura de la base de dades. En aquest fitxer estan definides totes les taules de la base de dades. Es defineixen així:

```
class GroupUser(models.Model):
    user = models.ForeignKey(CustomUser, on_delete=models.CASCADE)
    group = models.ForeignKey(Groups, on_delete=models.CASCADE)
    is_group_admin = models.BooleanField(default=False)

    class Meta:
        unique_together = ['user', 'group']
```

Figura 19: Exemple declaració d'un Model al fitxer models.py

Els models o classes es defineixen de forma que per exemple GroupUser sigui el nom de la taula en la base de dades. Dins de cada classe hi ha uns atributs com per exemple user, group i is_group_admin de la Figura 16.

Per identificar les claus foranes s'utilitza el models.ForeignKey(ModelReferenciat, opcions). Per exemple en cas de user, és clau forana del model CustomUser i quan s'elimini un CustomUser també s'eliminarà el GroupUser del qual n'és clau forana, utilitzant el mètode CASCADE.

Per definir els tipus dels atributs com per exemple is_group_admin que és un booleà, s'utilitza el mètode models."Type"Field().

Si s'ha d'afegir alguna restricció més sobre la taula, es defineix a la part de Meta. Com per exemple si hi ha dos claus que juntes han de ser úniques com està definit a l'exemple.

El fitxer **views.py** és on hi ha implementada tota la lògica de domini. Aquest fitxer és el que s'encarrega d'informar què s'ha de fer en qualsevol acció. Informa de quan s'ha d'agafar, modificar, crear o esborrar informació de la base de dades. Conté totes les crides possibles al sistema i s'encarrega de gestionar-ho. En aquest fitxer estan definits els mètodes POST, GET,

PUT i DELETE de les cada *url* corresponent, aquests mètodes corresponen a les accions CRUD (Create, Read, Update, Delete) .

Les crides a la API des del frontend es gestionen a través del fitxer **urls.py**. Conté de cada url quina part de la lògica de domini ha d'executar. Per exemple, una crida GET a http://{host}/ userList:

```
path('userList', ListUsersView.as_view(), name="user-list"),
```

Figura 20: Fitxer urls.py on s'indica a quina part del views.py correspon la url

Aquesta funció path indica que per la *url* /userList s'ha d'escollir la part de ListUsersView que està implementada al fitxer views.py.

```
class ListUsersView(generics.ListAPIView):
    serializer_class = UsersSerializer
    queryset = CustomUser.objects.all()
    permission_classes = (UsersPermission,)

    def get(self, request, *args, **kwargs):
        try:
            queryset = CustomUser.objects.all()
            serializer = UsersSerializer(queryset, many=True)
            return Response(serializer.data)
        except CustomUser.DoesNotExist:
            return Response(
                data={
                    "message": "Not found"
                },
                status=status.HTTP_404_NOT_FOUND
            )
```

Figura 21: Mètode GET de ListUsersView al fitxer views.py

Per tant, amb aquesta crida s'executarà aquest codi que retornarà una serialització de la llista d'usuaris. Per fer aquesta serialització actuarà el fitxer **serializers.py**. La serialització és un mecanisme que té Django per "traduir" els models de Django en altres formats, com per exemple JSON en el cas de la meva aplicació.

```
class UsersSerializer(serializers.ModelSerializer):
    class Meta:
        model = CustomUser
        fields = ("phone_number", "name", "surname", "country")
```

Figura 22: UserSerializer en el fitxer serializer.py

Aquest seria un exemple del UserSerializer de la meva aplicació, on model indica el model que estàs serialitzant i *fields*, els atributs que vols serialitzar del model. Si vols rebre atributs adicionals, s'ha d'indicar a través d'un mètode get dins del *serializer*.

Per tant, finalment l'usuari veurà a la seva pantalla un objecte JSON amb una llista de tots els usuaris de l'aplicació. El format JSON es basa en parelles clau valor.

Figura 23: Exemple CustomUser en format JSON

12.1.2. Autenticació

A continuació s'explicarà com el backend de l'aplicació genera el *token* que el frontend emmagatzemarà i utilitzarà.

Per crear el *token* s'ha utilitzat la llibreria TokenAuthentication de Django REST Framework. DRF ja té aquesta llibreria incorporada per tant, ja té preparada la Base de Dades per tenir una taula amb els t*okens* dels usuaris. Aquest *token* es crearà amb el registre de l'usuari, és a dir quan es vulgui fer un POST a http://{host}/UserList.

Serà una seqüència de número i lletres com aquesta:

```
f078fc6b1ffcb33c3317328149c24b504ad281b6
```

Figura 24: Exemple de *token*

Per tal d'utilitzar-lo s'haura d'afegir als Headers de la crida d'aquesta forma:

GET http://{host}/userList
Headers:
Authorization: Token f078fc6b1ffcb33c3317328149c24b504ad281b6

Aquesta autorització serà necessària per a totes les crides a l'API excepte el POST de /userList, ja que serà on es crearà. Per tal de fer això s'ha definit un fitxer de permisos on s'explica quan no serà necessari per *l'endpoint* /userList.

```python
class UsersPermission(permissions.BasePermission):

    def has_permission(self, request, view):
        if request.method == 'POST':
            return True
        else:
            return request.user.is_authenticated
```

Figura 25: Fitxer de permisos en el cas de /userList

En tots els altres casos, Django REST comprovarà els permisos que s'han declarat a cada usuari que per defecte seran que simplement estigui autenticat. Per fer això s'ha de declarar a un fitxer *settings* quins seran els permisos per defecte.

```python
REST_FRAMEWORK = {
    # Use Django's standard `django.contrib.auth` permissions,
    # or allow read-only access for unauthenticated users.
    'DEFAULT_PERMISSION_CLASSES': (
        'rest_framework.permissions.IsAuthenticated',
    ),
    'DEFAULT_AUTHENTICATION_CLASSES': (
        'rest_framework.authentication.TokenAuthentication',
    ),
```

Figura 26: Fitxer de configuració dels permisos i autenticació

En el cas que un usuari no estigui autenticat es retornarà l'error 401, Unauthorized amb el missatge: "Authentication credentials were not provided."

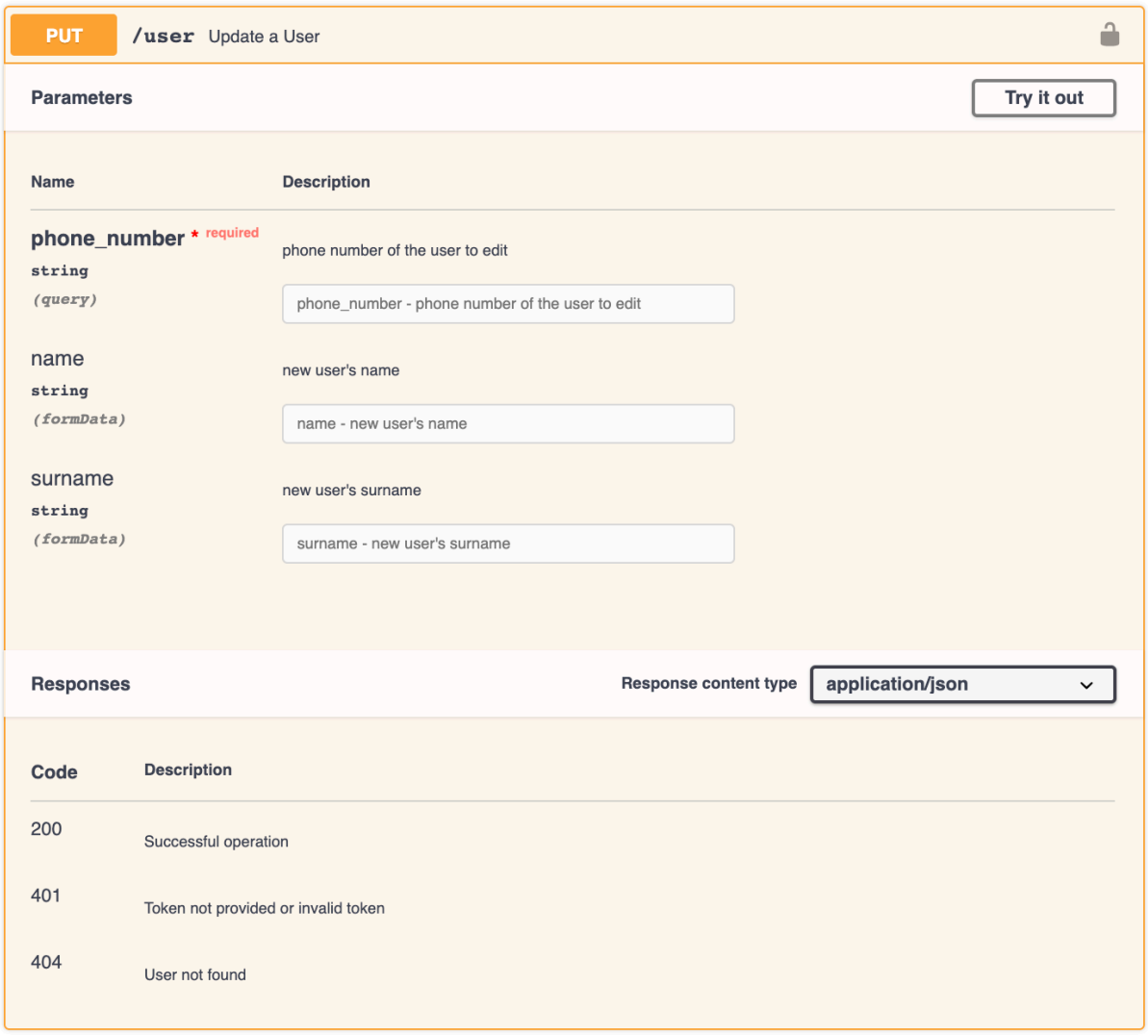
12.1.3. Endpoints

Els *endpoints* són les URLs d'una API que responen a una petició. Aquestes URLs contenen uns mètodes als que responen els *endpoints* i un *endpoint* pot respondre a més d'un mètode.

Els *endpoints* seran a través dels quals es modificarà o es llegirà el contingut de la Base de Dades. Els he documentat amb una eina que s'anomena Swagger.

Swagger és una eina que serveix per documentar API RESTs de forma que es poden distrubuir els *endpoints* per categories i dins d'aquestes categories queden plasmats amb la seva URL i el seu mètode. Si es prem a sobre d'un mètode d'un *endpoint* es pot veure amb més detall la crida, com per exemple els parametres que accepta aquesta crida, quins en són obligatoris, etc.

A més Swagger no només serveix per documentar, sinó que també serveix per testejar, ja que et deixa introduir els paràmetres que tu vols i et permet realitzar la crida a la URL indicada.



Alguns *endpoints* d'usuaris al swagger quedarien així:

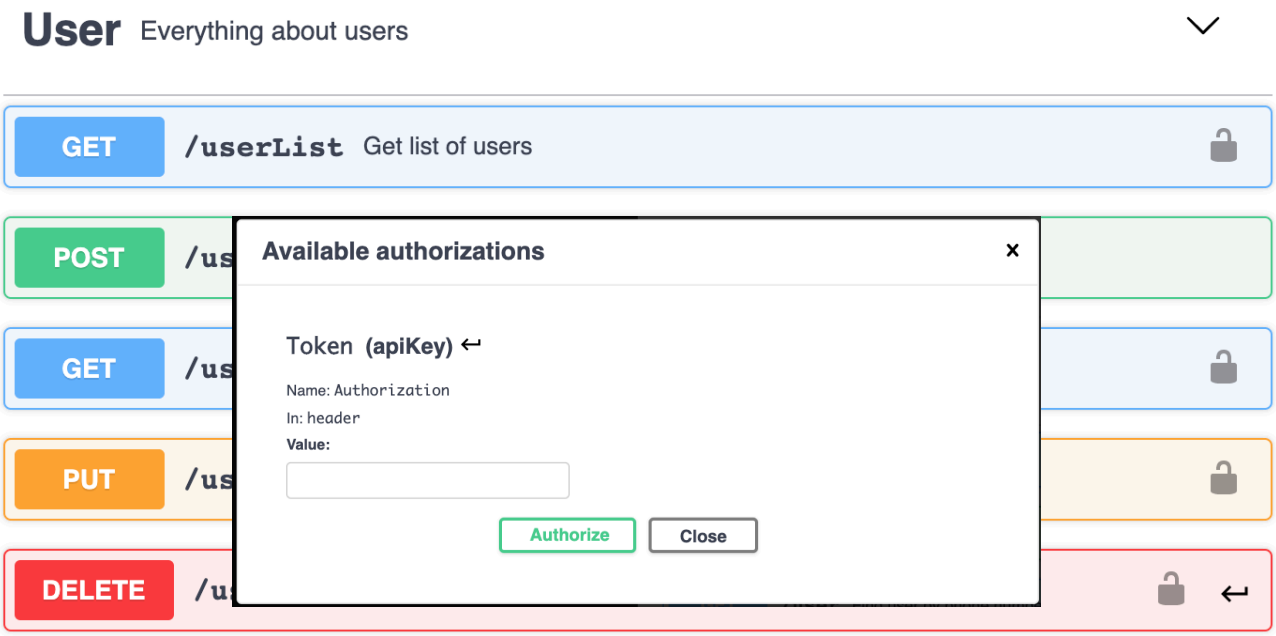


Figura 29: Modal per introduir el *token*

Figura 27: Document Swagger alguns endpoints User

Si volem veure en detall algun *endpoint,* es torna un desplegable que dóna informació sobre els paràmetres i les possibles respostes:

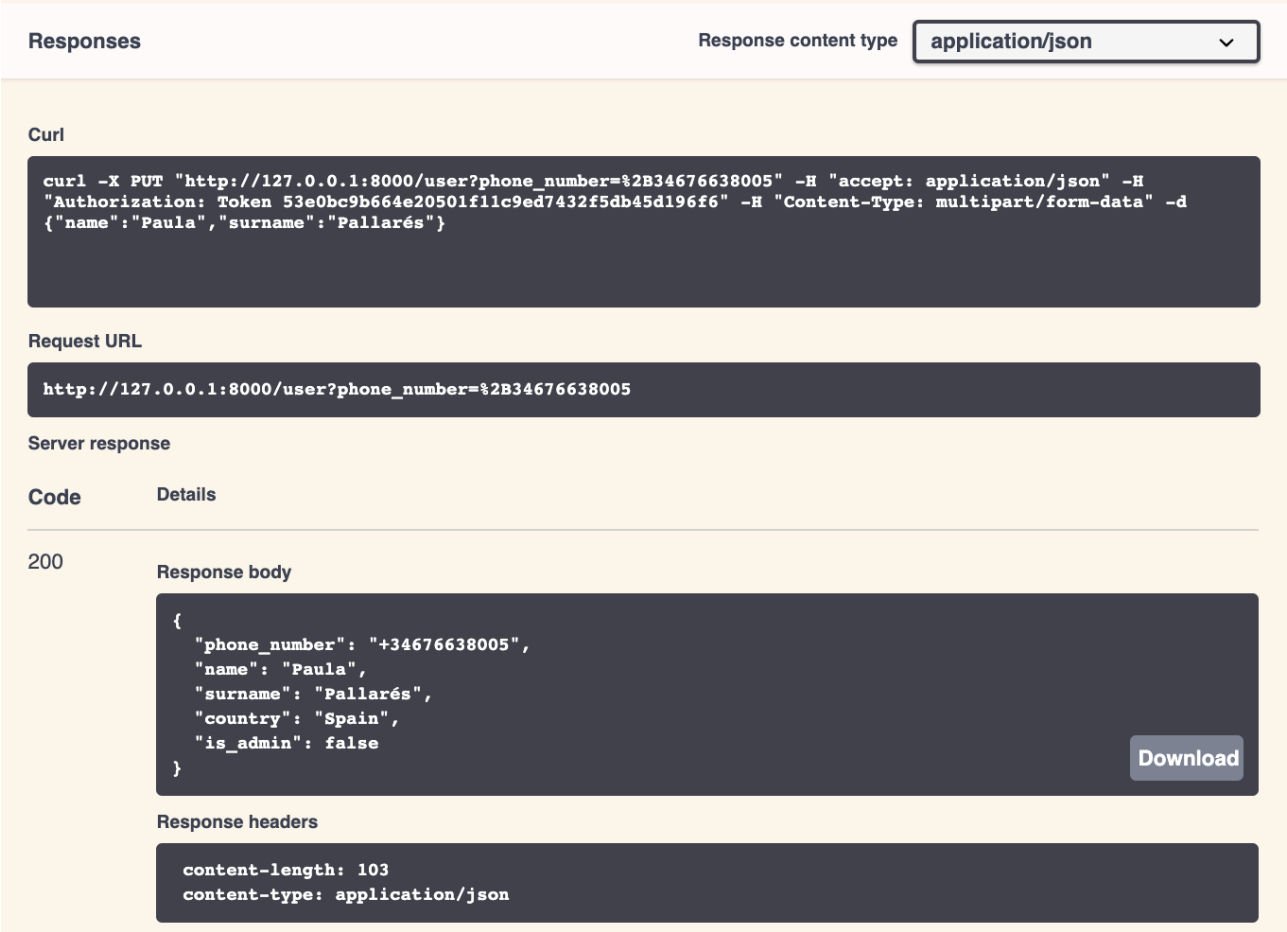
Figura 28: Detall de la crida PUT /user

Per provar la crida primer l'usuari s'hauria d'introduir el *token* prement la icona del cadenat, on s'obriria un modal com aquest:
Un cop introduït, s'hauria d'omplir el camp de phone_number, ja que és obligatori. Aquest camp formaria part de la *query* i indicaria el número de telèfon de l'usuari que es vol editar.

Finalment, segons les necessitats, s'hauria d'emplenar o bé el nom, o bé el cognom, o ambdues coses per editar l'usuari. Un cop emplenats s'hauria de prémer el botó *execute* i la resposta de la crida quedaria així:

Figura 30: Resposta a la crida PUT de /user

Per més informació sobre altres endpoints caldria anar a https://app.swaggerhub.com/apis/pauleta_6/HizzyAPIREST/1.0.3, on estan documentats amb detall tots els *endpoints* que hi ha a l'aplicació.



12.1.4. Llibreries utilitzades al backend

A continuació s'explicaran les llibreries que s'han utilitzat per la implementació del backend.

S'han utilitzat tres llibreries principalment. La primera es tracta d'una llibreria anomenada **django-cors-headers** [24], que s'ha utilitzat per poder documentar i testejar la API a través de Swagger. Aquesta llibreria habilita Cross-Origin Rosource Sharing (CORS), que això significa que els teus recursos podran ser accedits per altres dominis, com per exemple Swagger.

La segona llibreria que s'ha utilitzat és **django-events-rest-framework** [25]. Aquesta llibreria s'encarrega de tota la gestió interna de les dates en l'aplicació, ja que integra un calendari. Per Hizzy, un dels punts clau és que es puguin fer esdeveniments cada setmana com per exemple crear el rànquing setmanal i tornar a planificar la setmana que ve segons el mètode escollit. Per tant, aquesta llibreria és molt important, ja que és la que s'encarrega de crear esdeveniments que es repeteixin cada setmana. I a principi de setmana resetejarà el rànquing setmanal.

12.1.5. Deploy a Heroku

Heroku [26] és una plataforma Cloud que ofereix el servei de penjar la teva API a la seva plataforma a canvi d'una URL. Això permet que puguis utilitzar la teva API REST a través d'aquella

U R L i n o `import django_heroku` `django_heroku.settings(locals())` localment. És a dir, que ho p u g u i n utilitzar altres dispositius fora de l'àmbit local de l'ordinador.

Heroku té un repositori intern que es pot enllaçar amb el GitHub. Aquest repositori servirà a Heroku per tenir el codi de la API i utilitzar-lo per les crides al sistema.

Per tant, entenem deployament com aquest procés de penjar el codi a Heroku Cloud Platform per obtenir una URL.

Per tal de fer aquest deployament primer s'ha de crear un repositori a Heroku que estigui enllaçat a una URL. Després es penja el codi del repositori de GitHub a Heroku amb la comanda "git push heroku master".

Després s'ha de definir un Procfile, que és un fitxer de text que es situarà a la carpeta root del

```
web: gunicorn gettingstarted.wsgi --log-file -
```

projecte
per especificar quina comanda s'ha d'executar quan s'obre l'app.

Figura 31: Fitxer Procfile

Aquesta comanda és important perquè indicarà que el tipus de procés serà HTTP.

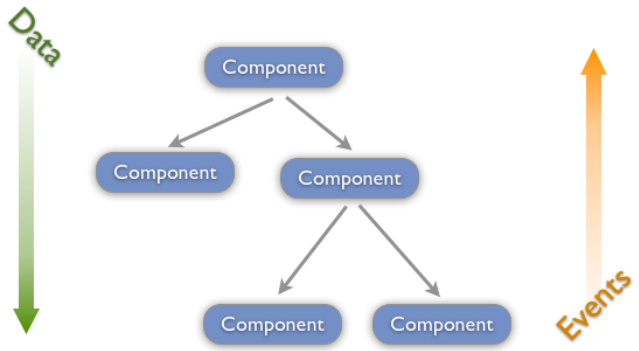
A continuació s'ha d'afegir al projecte dues llibreries més:

- **django-heroku** [36]. És la llibreria que permet deployar projectes Django a Heroku.
- **gunicorn** [37]. És la llibreria que permet indicar que el tipus de procés serà HTTP.

Després d'ha de canviar el fitxer **settings.py**, on s'afegirà aquest import i aquesta línia de codi:

Figures 32 i 33: Import i línia de codi al fitxer settings.py

Això serveix per indicar quina base de dades ha d'agafar la URL.



Finalment s'ha d'afegir al mateix fitxer la URL que s'ha obtingut per penjar el projecte a Heroku Cloud Platform. S'afegeix a ALLOWED_HOSTS.

Un cop realitzats tots aquests passos es torna a penjar al repositori els canvis realitzats i ja tenim Hizzy deployat a Heroku.

12.2. Frontend

En aquesta secció s'explicaran les característiques de la implementació del frontend de l'aplicació.

12.2.1. React Native

El llenguatge escollit per programar la part de frontend de l'aplicació és React Native [27]. React Native és un *framework* de JavaScript que permet crear aplicacions iOS i Android. Està basat en React, la llibreria JavaScript de Facebook per construir interfícies d'usuari, però en comptes d'anar destinat a un navegador, React Native només va destinat a plataformes mòbils.

Es va decidir aquest llenguatge, ja que comparteix codi tant per iOS com per Android, cosa que fa més fàcil per al programador no haver de programar codi per dues plataformes diferents i amb llenguatges de programació diferents. A més, la programadora ja tenia coneixements previs.

React Native és un llenguatge que utilitza els components natius dels telèfons mòbil, és a dir, ja compta amb components com el *keyboard*, l'*status bar* del telèfon, etc. A més a més, com que és un llenguatge que està basat en classes i components, es poden afegir més components i personalitzar-los al teu gust.

Els components poden tenir més components dins, és a dir que està basat en una arquitectura de components utilitzant un patró Parent to Child. Aquest patró s'encarrega de comunicar al component pare amb tots els seus fills de forma que els fills poden accedir a les propietats del pare. Mentre que el pare rep esdeveniments del fill mitjançant callbacks.

Figura 34: Estructura de components

12.2.2. Estructura del projecte

En aquest apartat s'expliquen les carpetes en les quals està estructurat el projecte:

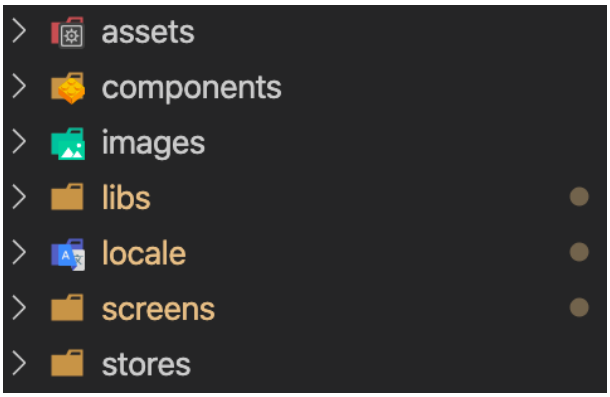


Figura 35: Carpetes del projecte

- **Assets**: És la carpeta que conté els fitxers que no formen part de l'aplicació però el codi necessita en *run time*. Com per exemple el tipus de font.

- **Components**: Són tots els components personalitzats creats manualment per la programadora que necessitarà per les pantalles de l'aplicació.

- **Images**: És la carpeta que conté totes les imatges de l'aplicació. Com per exemple les imatges de les banderes dels països.

- **Libs**: És la carpeta que conté els fitxers de configuració, com per exemple el fitxer que escull en quin llenguatge ha d'estar l'aplicació. A més també inclou el fitxer on estan totes les crides de l'API.

- **Locale**: En aquesta carpeta hi ha totes les traduccions en català, castellà i anglès de totes les frases que conté l'aplicació.

- **Screens**: Són totes les pantalles que té l'aplicació.

- **Stores**: Les *stores* són arxius d'emmagatzematge on estan guardades totes les dades generals de l'aplicació. Per exemple les dades de l'usuari que es necessiten en gairebé totes les pantalles. També pot emmagatzemar accions que també necessiten utilitzar-se en bastants llocs

i les quals no necessiten informació de la pantalla per executar-se.

La carpeta *screens* també està dividida en subcarpetes per cada una de les pantalles. Dins de cada subcarpeta hi ha dos fitxers, el fitxer on està programat el codi de la pantalla i el fitxer dels estils de la pantalla.

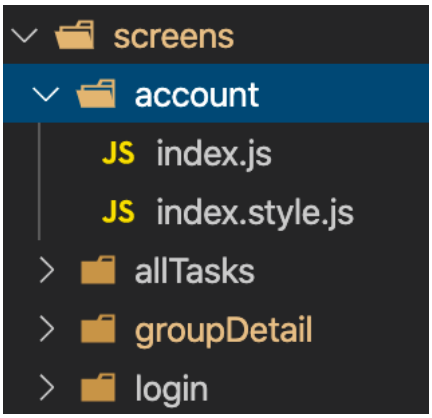


Figura 36: Carpeta screen amb unes quantes pantalles de l'aplicació

Finalment, hi ha un fitxer anomenat baseApp que serà la base de l'aplicació. Aquest fitxer contindrà la pila de navegació i serà l'encarregat a renderitzar totes les pantalles.

12.2.3. Autenticació

Per autenticar un usuari en la part de frontend s'utilitzarà un Login que també farà la funció de registre per aquells usuaris nous a l'aplicació.

Però abans d'explicar el funcionament, s'ha d'entendre què és el localStorage [28]. El localStorage és una memòria local de l'aplicació que emmagatzema dades que no tenen data d'expiració. Això significa que si un usuari tanca l'aplicació, les dades del localStorage persistiran igualment.

Aquesta tècnica s'utilitza amb la majoria d'usuaris per emmagatzemar el *token* al localStorage. D'aquesta forma es pot distingir si un usuari ja està loguejat a l'aplicació i així no haver de demanar que es loguegi cada cop que tanqui l'aplicació. El que també fan moltes aplicacions és que el *token* tingui una data de caducitat. Això farà que donat un cert temps que no s'obri l'aplicació l'usuari estigui forçat a tornar a loguejar-se. En el cas de Hizzy el *token* és permanent.

El funcionament és el següent:

Inici de sessió

1. L'usuari introdueix el seu prefix i número de telèfon a la pantalla de Login.
2. El sistema fa una petició al backend amb les credencials de l'usuari.

a. Si l'usuari ja existeix, el backend retornarà el *token* de l'usuari.

b. Si l'usuari no existeix, el backend retornarà un nou *token* per l'usuari.

3. El sistema emmagatzema el *token* al localStorage.
4. El sistema navega

a. Si l'usuari ja existeix, el sistema navega a la pantalla principal amb la informació dels grups de l'usuari.

b. Si no existeix, el sistema navega a la pantalla *d'account* per acabar d'omplir les dades que fan falta.

5. El *token* emmagatzemat s'envia juntament amb totes les peticions.

Tancament de sessió

El sistema deixa el *token* emmagatzemat al localStorage.

Eliminació del compte o de l'aplicació

El sistema elimina el *token* de la memòria local.

12.2.4. Llibreries utilitzades al frontend

Mobx

Una de les llibreries utilitzades més important és **mobx** que ja està explicada anteriorment a l'apartat 11.6, en la part del patró observador.

React Navigation

Una altra llibreria principal que s'utilitza és **react-navigation** [29]. Aquesta llibreria s'utilitza per a la navegació entre pantalles. Ofereix tres tipus de navegació:

- **Stack Navigation**: Aquest tipus de navegació és la més comuna en les aplicacions, ja que és una pila on estan definides totes les pantalles amb la seva ruta. Això permet que es pugui navegar des de qualsevol pantalla a qualsevol altra només indicant el nom de la ruta on es vol anar. Aquest tipus de navegació també et permet definir una ruta inicial. Per diferenciar-la de les altres navegacions, en aquesta les pantalles apareixen des de la part inferior de la pantalla del telèfon.
- **Drawer Navigation**: Aquest tipus de navegació s'utilitza quan es vol utilitzar un menú, com per exemple un menú lateral d'una aplicació. Per definir les pantalles d'un menú lateral també es fa a través d'una pila que contindrà el nom de les pantalles amb el nom de la ruta. Des d'un menú lateral es podrà navegar a les pantalles definides a la pila indicant també el nom de la ruta on es vol anar. Aquestes pantalles apareixen des dels laterals de la pantalla.

- **Tabs Navigation**: Aquest tipus de navegació s'utilitza quan es vol tenir una barra fixa de navegació a la part inferior de la pantalla. Aquesta navegació serveix per definir aquelles funcionalitats principals que es vol que estiguin més a la vista per anar més ràpidament per l'usuari. També s'utilitza una pila per definir les pantalles amb les seves rutes corresponents.

Hizzy utilitza una mescla de les tres navegacions, ja que una pila pot referenciar-ne a una altra. La Tabs Navigation la utilitza amb una barra de navegació que es trobarà en algunes pantalles. Aquesta barra és així:

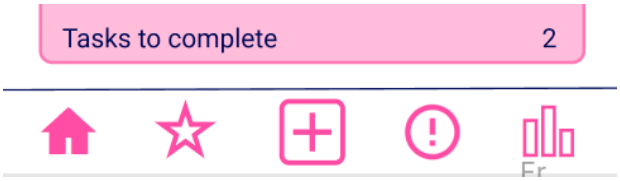


Figura 37: Barra de navegació Hizzy

La referenciació entre piles quedaria així:

```
const mainTabs = createBottomTabNavigator(
  {
    Main: {
      screen: mainStack,
      navigationOptions: {
        tabBarIcon: ({ tintColor }) =>
          tintColor === "#F25CA3" ? (
            <Home></Home>
          ) : (
            <HomeOutline></HomeOutline>
          )
      }
    },
    Rate: {
      screen: RateScreen,
      navigationOptions: {
        tabBarIcon: ({ tintColor }) =>
          tintColor === "#F25CA3" ? (
            <Star></Star>
          ) : (
            <StarOutline></StarOutline>
          )
      }
    }
  },
```

Figura 38: Exemple de pila de Tab Navigation

Com podem veure en la Figura 32, hi ha una part de la declaració de la barra de navegació. Per exemple la ruta Rate seria la pantalla de valoració als companys, mentre que Main fa referència a una altra pila que s'anomena mainStack.

Figura 39: Exemple de pila

```
const mainStack = createStackNavigator(
  {
    Main: {
      screen: drawerStack,
      navigationOptions
    },
    Login: {
      screen: LoginScreen,
      navigationOptions: {
        tabBarVisible: false
      }
    },
```

Stack Navigation

En la Figura 33 hi ha la pila anomenada mainStack que utilitza Stack Navigation. En aquesta pila estan definides totes aquelles pantalles que s'han d'accedir des de qualsevol lloc que no sigui el menú lateral o la barra de navegació. Es poden afegir opcions de navegació per cada pantalla, com per exemple que no es vegi la barra de navegació en la pantalla de Login. La ruta Main fa referència a la drawerStack que és la pila del menú lateral Drawer Navigation.

```
const drawerStack = createStackNavigator(
  {
    Main: {
      screen: drawer,
      navigationOptions
    },
    Account: {
      screen: AccountScreen
    }
  },
```

Figura 40: Exemple de pila Drawer Navigation

En la Figura 34 hi ha la pila anomenada drawerStack que utilitza Drawer Navigation. En aquesta pila estan definides aquelles pantalles que s'accedeixen des del menú lateral. Per exemple la ruta Account va a la pantalla del perfil i la ruta Main referència a la pantalla principal amb el component *drawer*.

Axios

La llibreria que s'utilitza per realitzar crides a l'API és **axios** [30]. Axios és un client HTTP basat en promeses, això significa que s'espera fins a rebre una resposta del servidor per transferir les dades a la interfície. Per tant, el que fa axios és realitzar les crides HTTP al servidor i rebre les dades i transferir-les a la interfície.

Liner Gradient

Una de les llibreries que s'utilitza per als colors de l'aplicació és **react-native-linear-gradient** [31]. Aquesta llibreria s'utilitza per aconseguir l'efecte gradient en els colors. Per exemple si ens fixem

en el botó de confirmar, es pot apreciar que per l'esquerra del botó és d'un blau més clar que per la dreta del botó. Aquest efecte s'aconsegueix utilitzant aquesta llibreria.

Figura 41: Botó de confirmació de l'aplicació

SVG

Una de les funcionalitats principals del llenguatge React Native és que es poden crear components. De la mateixa forma passa amb imatges *svg*, és a dir, React Native permet crear components a partir d'una imatge amb extensió *svg*. Per tal transformar una imatge *svg* a component s'ha d'utilitzar la llibreria **react-native-svg** [32]. Aquesta transformació permetrà que una imatge *svg* es pugui utilitzar dins d'una pantalla com si fos un component més de la pantalla, al mateix nivell que un botó per exemple.

Moment

La llibreria que s'utilitza per transformar els formats de dates és **moment** [33]. Moment és una llibreria que s'utilitza per crear formats de dades a partir del format estàndard de representació d'una data. Aquest format es representa així: 2019-11-02T12:00:43Z. A partir d'aquest format, moment t'ofereix representar la data de moltes possibles maneres.

Dropdown Alert

La llibreria que s'utilitza per al control d'errors és **react-native-dropdownalert** [34]. Aquesta llibreria permetrà indicar a l'usuari quan ha fet algun error, o simplement se'l vol informar d'alguna acció que s'ha realitzat correctament com per exemple modificar correctament les seves dades del compte. Aquestes informacions/errors es mostraran a l'usuari en forma de *banners* com els de la Figura 36 i 37.

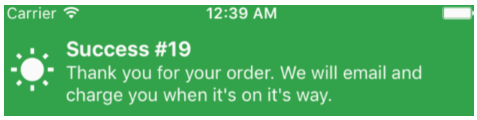


Figura 42: *Banner* de success

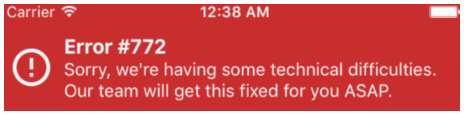


Figura 43: *Banner* d'error

Calendar

La llibreria utilitzada per escollir un dia del calendari quan es crea o s'edita una tasca és **react-native-calendars** [38]. Aquesta llibreria proporciona un calendari com el de la Figura 38 que serveix per seleccionar un dia. Per tant, per la data d'inici de la tasca i la data fi és una llibreria molt útil. A més, et permet afegir restriccions al calendari com per exemple que no es puguin seleccionar dates anteriors al dia d'avui.

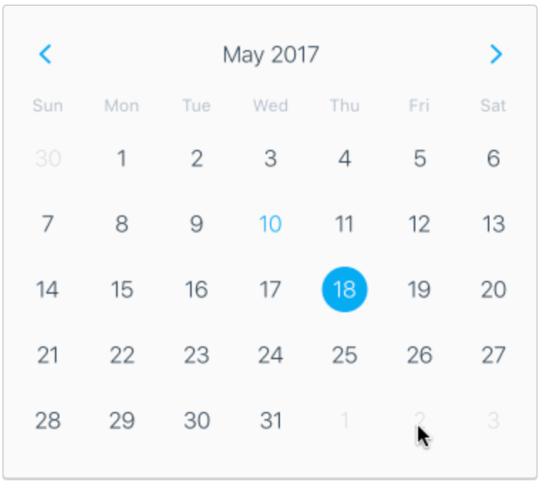


Figura 44: Exemple de calendari de la llibreria react-native-calendars

13. Proves

Les proves i el testing de l'aplicació s'han fet tan com pel *frontend* i el *backend*.

13.1. Frontend

Per la part de *frontend* s'han fet proves d'experiència d'usuari. En primer lloc, per part de la desenvolupadora del projecte i en segon lloc, amb les companyes de pis. S'han provat les funcionalitats desenvolupades de l'aplicació per la recerca d'errors o possibles millores. En el cas de trobar algun error se li ha notificat a la desenvolupadora i aquesta ho ha arreglat.

13.2. Backend

Per la part de *backend* s'han fet proves a través de l'eina Swagger, explicada anteriorment. Aquestes proves s'han realitzat manualment pensant amb els possibles errors que podria haver-hi i la resposta esperada per cada crida en cada moment. A més, els errors que pot donar la crida també estan documentats. La manera de testejar una crida és la següent:

1. Escollir l'*endpoint* que es vol testejar.
2. Pensar en totes les possibles respostes i els errors que pot generar aquell *endpoint*.
3. Anar a l'eina swagger i autenticar-se si l'*endpoint* requereix autenticació.
4. Prémer sobre el mètode que es vol testejar.
5. Emplenar els camps necessaris per testejar la crida.
6. Prémer el botó d'executar.
7. Comprovar el resultat.

D'aquesta manera s'han realitzat totes les proves dels *endpoints* de la API. A més durant el testeig del *frontend* també es fan proves del *backend* indirectament.

14. Planificació final

A continuació s'explicaran els canvis que hi ha hagut en la planificació del projecte respecte a la planificació inicial plantejada. I també es comentaran les iteracions del projecte.

14.1. Calendari

Hi ha dues fases que han patit modificacions, han estat la d'implementació i la de documentació. Aquestes dues fases estan dividides en subfases principals que quedaran d'aquesta forma:

Implementació:

- Preparació de l'entorn
- Implementació *backend*
- Implementació *frontend*
- *Testing* i *review*

Documentació:

- Documentació final
- Preparar defensa

Els canvis que hi ha hagut són la dedicació de més hores en la part d'implementació del *backend* i *frontend* i l'eliminació del manual d'usuari en la part de documentació.

El motiu pel qual ha passat això és perquè la part d'implementació ha acabat requerint més temps del que hauria i no ha donat temps a fer el manual d'usuari. Per tant, això ha afectat de la següent forma el projecte:

Tasca	Hores	Dependències
1. Tasques inicials (GEP)	**61h**	**-**
1.1. Definició de l'abast i contextualització	25h	-
1.2. Planificació temporal	8h	1.1
1.3. Estimació de costos	10h	1.2
1.4. Entrega final GEP	18h	1.3
2. Anàlisi i disseny	**85h**	**-**
2.1. Anàlisi requisits i funcionalitats	35h	1.1
2.2. Disseny arquitectura	15h	2.1
2.3. Disseny UML	10h	2.2
2.4. Disseny interfície	25h	2.3

Tasca	Hores	Dependències
3. Implementació	**280h**	**2**
3.1. Preparació entorn	10h	2.4
3.2. Implementació *backend*	80h	3.1
3.3. Implementació *frontend*	140h	3.1
3.4 *Testing* i *review*	50h	3.2 i 3.3
4. Documentació	**110h**	**-**
4.1. Documentació final	90h	-
4.3. Preparar defensa	20h	4.2
Total	**536h**	**-**

Taula 8: Durada de les tasques, planificació final

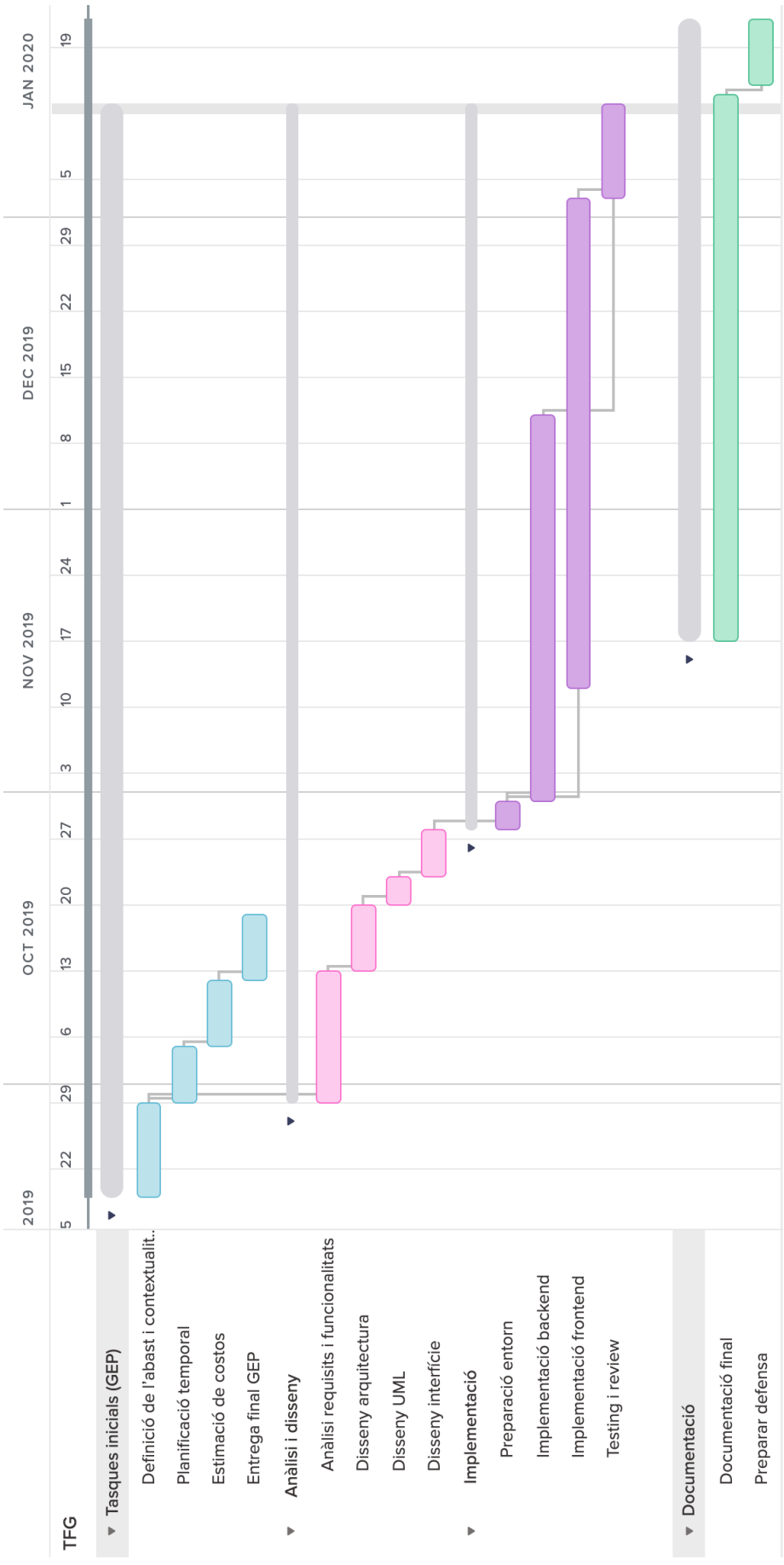


Figura 45: Diagrama de Gantt, planificació final

14.2. Gestió econòmica

Que hi hagi hagut canvis en la planificació final implica que també hi ha hagut canvis en la gestió econòmica del projecte. Com que només ha canviat la planificació temporal només afectarà els salaris dels recursos humans, però com que les 20h de la programadora del manual d'usuari se sumen a la implementació també per la part de la programadora el cost total del projecte quedaria igual. De totes maneres es modificarien els apartats de manera que queda així:

Tasca	**Cap de projecte**	**Analista**	**Dissenyadora**	**Programadora**	**Tester**	**Cost per tasca i fase**
Tasques inicials (GEP)	**61h**	**-**	**-**	**-**	**-**	**1647€**
Definició de l'abast	25h	-	-	-	-	675€
Planificació temporal	8h	-	-	-	-	216€
Estimació costos	10h	-	-	-	-	270€
Entrega final GEP	18h	-	-	-	-	486€
Anàlisi i disseny	**10h**	**25h**	**50h**	**-**	**-**	**1645€**
Anàlisi requisits	10h	25h	-	-	-	845€
Disseny arquitectura	-	-	15h	-	-	240€
Disseny UML	-	-	10h	-	-	160€
Disseny interfície	-	-	25h	-	-	400€
Implementació	**5h**	**-**	**-**	**195h**	**80h**	**4180€**
Preparació entorn	-	-	-	10h	-	150€
Implementació *backend*	-	-	-	65h	15h	1185€
Implementació *frontend*	-	-	-	120h	20h	2080€
Testing i *review*	5h	-	-	-	45h	765€
Documentació	**110h**	**-**	**-**	**20h**	**-**	**2970€**
Documentació final	90h	-	-	-	-	2430€
Preparar defensa	20h	-	-	-	-	540€
Hores per rol	**186h**	**25h**	**50h**	**195h**	**80h**	**-**
Cost per rol (sense seguretat social)	**5022€**	**575€**	**800€**	**2925€**	**1120€**	**10442€**
Cost per rol*1,35% seguretat social	**6779,7€**	**776,25€**	**1080€**	**3948,75€**	**1512€**	**14096,7€**

Taula 9: Costos per tasques, rols i hores dedicades, planificació final

14.3. Iteracions

La implementació d'aquesta aplicació s'ha dividit en iteracions. En total s'han fet 8 iteracions entre les quals es divideixen totes les funcionalitats de l'aplicació.

Iteració 1

Aquesta iteració ha servit per crear la base a partir de la qual construir l'aplicació. La iteració s'ha dividit en dues parts principals:

- **Creació base de dades**: Implementació dels models i les relacions entre ells. A partir d'aquí realitzar les migracions a la Base de Dades per crear les taules.

- **Creació d'alguns *endpoints***: Implementació dels *endpoints* sobre grups i usuaris.

Els *endpoints* que es van crear a la iteració serien /userList, /user, /groupList, /group/{groupId}, /group/{groupId}/user, /group/{groupId}/me i /group/{groupId}/admin amb els seus respectius mètodes CRUD.

Iteració 2

Durant la segona iteració s'han realitzat els *endpoints* relacionat amb les tasques i els mètodes amb els mètodes CRUD corresponents. Aquests *endpoints* són: /task, /task/{taskId}, /task/{taskId}/user, /method, /method/{methodId}, /method/{methodId}/task i /method/{methodId}/user. A més també s'han afegit els endpoints /group/{groupId}/method. A més, s'han documentat tots els *endpoints* fets fins al moment.

Iteració 3

Durant la iteració 3 s'ha començat a implementar la interfície, en concret, la pantalla de Login, la de pantalla de seleccionar el país, la Launcher Screen (que és la pantalla de càrrega quan s'obre una app), la pantalla Account i el Logout.

Per altra banda, també s'ha configurat l'esquema del projecte, la navegació entre pantalles i s'ha programat la comunicació entre el frontend i el backend mitjançant axios.

Iteració 4

A la quarta iteració s'ha continuat implementant la interfície. S'han implementat la pantalla principal, la de crear i editar grup, la pantalla d'informació del mètode i la de seleccionar el mètode. Per la part de l'API s'han començat a realitzar els *endpoints* relacionats amb les valoracions i puntuacions i els rànquings. Aquests *endpoints* són: /task/{taskId}/rate, /user/rankings, /user/ratings.

Iteració 5

A la quinta iteració s'ha continuat implementant la interfície. S'han implementat la pantalla de crear i editar una tasca, la pantalla de detall d'una tasca i la de seleccionar un membre. Per la part de la API s'han començat a realitzar els endpoints per les bonificacions i els comentaris: /user/bonifications, /user/bonifications/{bonificationId}, /task/{taskId}/comment, /task/{taskId}/comment/{commentId}

Iteració 6

A la sexta iteració s'ha acabat d'implementar el *backend*. Per acabar de fer-ho s'ha utilitzat la llibreria django-rest-events, esmentada anteriorment, per programar aquells esdeveniments del sistema. Per la part d'interfície s'ha implementat la pantalla de crear i editar un mètode,

Iteració 7

En aquesta iteració s'ha deployat el *backend* a Heroku. Per la part d'interfície s'han implementat les pantalles de seleccionar un grup, tasques pendents i la de seleccionar el dia de la setmana.

Iteració 8

En aquesta iteració s'han implementat les pantalles valoracions, rànquings, bonificacions, totes les tasques d'un grup i s'han afegit els comentaris a la interfície.

Durant totes les iteracions en les quals s'ha realitzat part de frontend, s'han anat fent les traduccions per als tres idiomes (català, castellà i anglès).

15.Lleis i regulacions

Aquest projecte es veu afectat per una sola llei, el Reglament General de Protecció de Dades (GDPR) [39].

Això és degut al fet que l'aplicació desenvolupada durant aquest projecte utilitza els números de telèfon dels usuaris per tal d'identificar-los.

La resta de funcionalitats de l'aplicació no es veuen afectades per cap llei, ja que tracten sempre amb dades internes de l'aplicació.

Per tal de complir la llei, les dades dels usuaris no s'utilitzaran per cap altre motiu que no sigui el de poder utilitzar correctament l'aplicació.

16. Informe de sostenibilitat

16.1. Dimensió econòmica

Pel que fa a la dimensió econòmica, les despeses previstes i el càlcul total del desenvolupament del projecte es troben explicats als apartats anteriors. El cost total del projecte serà de **12343,36€**.

Pel que fa a la vida útil del projecte, no hi ha cap aplicació que es dediqui exactament a fer el que ofereix Hizzy, però sí que es fa manualment com per exemple tenir materials com una pissarra o fulls i llistes per distribuir les tasques i fer un calendari. El meu projecte evitaria aquests costos als usuaris de l'aplicació, ja que tot estaria en format digital.

16.2. Dimensió ambiental

Aquest projecte s'ha realitzat amb l'energia d'un únic ordinador, que consumeix uns 0,5 kWh. Si multipliquem per les 536 hores que durarà el projecte, s'han consumit **268kW** totals.

Pel que fa al cost de llum i calefacció, entre altres, en desenvolupar-se a casa aquest consum acaba sent el mateix que si estigués a casa fent altres coses.

A més, cal tenir en compte que tot el projecte s'ha dut a terme de manera digital, per tant no s'han generat residus que sigui necessari reciclar.

Per altra banda, el portàtil i telèfons utilitzats en el projecte són de la meva propietat, i s'han anat utilitzant en altres projectes, de la mateixa forma que s'utilitzaran en un futur. A més, també es podrà reutilitzar codi d'altres projectes ja realitzats amb el mateix llenguatge de programació.

Pel que fa a la vida útil, com ja he comentat abans, el meu projecte ha evitat els costos materials de manera que no es gastarien fulls o altres materials contaminants, per tant no deixaria cap petjada ecològica.

16.3. Dimensió social

Pel que fa a la dimensió social, crec que aquest projecte m'ha fet ser conscient de tots els problemes que pot tenir la societat que comparteix habitatge i intentar solucionar-los ha sigut un repte que m'ha ajudat també a créixer com a persona. M'ha fet empatitzar amb la gent afectada per aquesta problemàtica.

Aquest projecte garantirà un compromís d'aquelles persones que comparteixen pis de forma que ara ja no hi ha cap excusa de "no m'havia recordat", "he perdut els papers", "com no hi havia res pactat no ho he fet", etc. Fent així que les persones que conviuen en un mateix habitatge puguin estar tranquil·les i més contentes en les seves vivendes i amb aquelles persones que finalment acaben passant la majoria del seu temps.

Així doncs, crec que existeix una necessitat real del projecte, ja que és una problemàtica actual que amb la meva solució aporta benestar a la societat afectada.

17. Conclusions

Després d'haver explicat els detalls importants sobre el disseny i desenvolupament de Hizzy, per a les conclusions s'analitzarà si s'han assolit els objectius marcats a l'inici del projecte i si s'han assolit les competències tècniques definides a l'inici. Per acabar, es planteja un possible futur pel projecte i per últim, es farà una reflexió sobre l'aprenentatge personal de l'estudiant durant el desenvolupament del projecte.

17.1. Assoliment dels objectius del projecte

Els diferents objectius principals del projecte a l'apartat 5.1.Objectius s'han assolit tots de manera correcta. Pel que fa als subobjectius només hi ha un que no s'ha pogut assolir que ha sigut el tema de notificacions i enllaçar el calendari.

L'objectiu clau del projecte ha sigut: *dissenyar i desenvolupar una aplicació mòbil per ajudar a organitzar-se dins d'un pis compartit i assegurar el compromís a través de la gamificació*. Després de tot el que s'ha explicat en aquest document es pot afirmar que l'aplicació desenvolupada compleix amb aquest objectiu.

El fet d'haver centrat el disseny de la interfície en la simplicitat i la usabilitat, implica que els usuaris puguin utilitzar l'aplicació d'una forma fàcil i intuïtiva. A més, el fet d'haver utilitzat stakeholders reals i haver centrat la problemàtica en un àmbit en el qual em trobo a dins, ha convertit la solució en una solució real per utilitzar.

En conclusió, es pot afirmar finalment que l'objectiu que es volia aconseguir amb aquest projecte ha estat complet.

17.2. Competències tècniques

Les competències tècniques que s'han inclòs en aquest projecte són les següents:

- **CES1.3: Identificar, avaluar i gestionar els riscos potencials associats a la construcció de software que es poguessin presentar.**
  Aquesta competència tècnica es correspon amb el procés dut a terme durant el plantejament del projecte d'analitzar els riscos que podien aparèixer al llarg del desenvolupament d'aquest.

- **CES1.5: Especificar, dissenyar, implementar i avaluar bases de dades.**
  Aquesta competència tècnica s'ha realitzat a l'hora de dissenyar i implementar la base de dades de l'aplicació.

- **CES1.7: Controlar la qualitat i dissenyar proves en la producció de software.**
  Aquesta competència tècnica s'ha treballat durant les fases de testing de les diferents iteracions del projecte. Aquestes proves estan explicades a l'apartat 13.Proves.

- **CES2.1: Definir i gestionar els requisits d'un sistema software.**
  El compliment d'aquesta competència tècnica es correspon amb una de les parts més importants del projecte. S'ha assolit a causa de les reunions i entrevistes amb les companyes de pis i a més, de collita pròpia. Després s'han gestionat de la millor manera possible.

17.3. Treball futur

Aquesta aplicació podria tenir futur, ja que no existeix res igual al mercat i soluciona una problemàtica actual en la qual es troben moltes persones. De cara a un futur es podria plantejar de treure l'aplicació al mercat per poder oferir el servei a les persones que ho necessitin, afegint les funcionalitats que han quedat per fer. Aquestes funcionalitats són les notificacions i l'enllaç a un calendari extern.

A més també s'haurien d'afegir algunes millores i explicacions perquè la gent acabés d'entendre què ofereix aquesta aplicació.

Posteriorment es completaria amb noves funcionalitats de manera que l'aplicació es pugui adaptar a les necessitats dels usuaris.

17.4. Conclusions personals

Aquest treball ha estat molt gratificant en l'àmbit personal. Primerament perquè és el primer treball que realitzo jo sola. Considero que al proposar-lo pensava que seria molt més fàcil realitzar-lo, però un cop vist el que realment era m'he adonat que no era gens fàcil. Al final, dins d'aquest treball he acabat fent molts rols: cap de projecte, programadora, tester, traductora i dissenyadora.

Com que mentre realitzava el projecte he estat fent pràctiques d'empresa a Reby Rides S.L., m'he adonat del que realment comporta crear una aplicació per llençar-la al mercat i oferir un servei a la societat. M'he adonat que tots els rols que jo he fet en un projecte com aquest, en una empresa són persones diferents amb contractes laborals de 40 hores setmanals. Per tant, considero que estic molt satisfeta amb la feina feta.

A més, ha sigut un repte que m'ha agradat molt intentar assolir, ja que és una problemàtica en la qual jo mateixa estic inclosa i m'ha agradat desenvolupar una solució al respecte.

He après molt durant el projecte i he tingut l'oportunitat d'aprendre i millorar com a programadora. Tota aquesta expriència m'ha ajudat a millorar els meus coneixements sobre l'Enginyeria del Software.

18. Referències

[1] Yolanda Garcia. (Sep 13, 2019). La edad media para compartir piso en España aumenta de 29 a 34 años: el no poder pagar un alquiler solo es la principal razón. from: https://www.eleconomista.es/vivienda/noticias/10082553/09/19/La-edad-media-para-compartir-piso-en-Espana-aumenta-de-29-a-34-anos-en-apenas-un-ano-el-no-poder-pagar-un-alquiler-solo-es-la-principal-razon.html

[2] Pepe Raga. (Aug 27, 2019). Cuando compartir piso ya no es una práctica exclusiva de estudiantes. from: https://www.eldiario.es/economia/compartir-piso-practica-exclusiva-estudiantes_0_935806568.html

[3] Facultat d'Informàtica de Barcelona. Normativa TFG GEI Final. from: https://www.fib.upc.edu/sites/fib/files/documents/estudis/normativa-tfg-gei-final.pdf

[4] Pelayo Alvarez. (Feb 17, 2016). Siete aplicaciones imprescindibles para compartir piso. from: https://www.elmundo.es/f5/2016/02/17/56c47c3546163f0e588b45d4.html

[5] Chelsey Pippin. (Feb 03, 2015). 16 Apps Everyone Who Lives In A Shared House Needs. from: https://www.buzzfeed.com/chelseypippin/houseshare-apps

[6] Find your next roommate with Badi. from: https://www.shbarcelona.com/blog/en/badi/

[7] Habitoom. from: http://www.habitoom.com/press/detail/50

[8] Dividir gastos con amigos. :: Splitwise. from: https://www.splitwise.com/index

[9] Tricount - FAQ - How does Tricount work? from: https://www.tricount.com/en/how-tricount-works

[10] Picniic | The home hub platform. from: https://picniic.com

[11] Tody. from: http://todyapp.com

[12] Robertson S Robertson J. (2006). Mastering the requirements process.

[13] Online Gantt Chart Software | TeamGantt. from: https://www.teamgantt.com/

[14] Hays. (2019). Guía del Mercado Laboral 2019. from: https://guiasalarial.hays.es/

[15] iPhone - Comparar modelos - Apple (ES). from: https://www.apple.com/es/iphone/compare/

[16] Xiaomi Redmi 6A 2+16GB 5.45"" Negro - Teléfono móvil libre - Los mejores precios | Fnac. from: https://www.fnac.es/mp6420376/Xiaomi-Redmi-6A-2-16GB-5-45-Negro/w-4

[17] Aida Jurado. (Jan 02, 2014). Què és la gamificació? from: https://www.viaempresa.cat/l-expert/que-es-la-gamificacio_4027_102.html

[18] HTTP response status codes - HTTP | MDN. from: https://developer.mozilla.org/en-US/docs/Web/HTTP/Status

[19] Figma: the collaborative interface design tool. from: https://www.figma.com/

[20] Valentin Kononov. (Jan 28, 2019). State Management — how we do it with MobX in Mobile App. from: https://medium.com/akveo-engineering/state-management-how-we-do-it-with-mobx-in-mobile-app-c381d5b3cca1

[21] Higher-Order Components – React. from: https://reactjs.org/docs/higher-order-components.html

[22] Home - Django REST framework. from: https://www.django-rest-framework.org/

[23] About SQLite. from: https://www.sqlite.org/about.html

[24] GitHub - adamchainz/django-cors-headers: Django app for handling the server headers required for Cross-Origin Resource Sharing (CORS). from: https://github.com/adamchainz/django-cors-headers

[25] GitHub - aswinzz/django-events-rest-framework: Out of the box package for setting up events using django rest framework along with easy integration to fullcalendar. from: https://github.com/aswinzz/django-events-rest-framework

[26] Deploying with Git | Heroku Dev Center. from: https://devcenter.heroku.com/articles/git

[27] React Native · A framework for building native apps using React. from: https://facebook.github.io/react-native/

[28] LogRocket Blog (July 17, 2018). LocalStorage in JavaScript. from: https://blog.logrocket.com/the-complete-guide-to-using-localstorage-in-javascript-apps-ba44edb53a36/

[29] React Navigation · Routing and navigation for your React Native apps. from: https://reactnavigation.org/

[30] GitHub - axios/axios: Promise based HTTP client for the browser and node.js. from: https://github.com/axios/axios

[31] GitHub - react-native-community/react-native-linear-gradient: A <LinearGradient /> component for react-native. from: https://github.com/react-native-community/react-native-linear-gradient

[32] GitHub - react-native-community/react-native-svg: SVG library for React Native, React Native Web, and plain React web projects. from: https://github.com/react-native-community/react-native-svg

[33] Moment.js | Home. from: https://momentjs.com/

[34] GitHub - testshallpass/react-native-dropdownalert: A simple alert to notify users about new chat messages, something went wrong or everything is ok. from: https://github.com/testshallpass/react-native-dropdownalert

[35] Robertson, Suzanne. (2013). Mastering the requirements process : getting requirements right / Suzanne Robertson, James Robertson.

[36] django-heroku · PyPI. from: https://pypi.org/project/django-heroku/

[37] gunicorn · PyPI. from: https://pypi.org/project/gunicorn/

[38] GitHub - wix/react-native-calendars: React Native Calendar Components. from: https://github.com/wix/react-native-calendars

[39] General Data Protection Regulation (GDPR) – Official Legal Text. from: https://gdpr-info.eu/

[40] Mike Cohn. (2006). Agile estimating and planning /Mike Cohn.

[41] Mike Cohn. (2004). User stories applied: for agile software development /Mike Cohn.

[42] Shklar, Leon. (2009). Web application architecture : principles, protocols, and practices / Leon Shklar and Richard Rosen.

[43] Teorey, Toby J. (2011). Database modeling and design : logical design / Toby Teorey